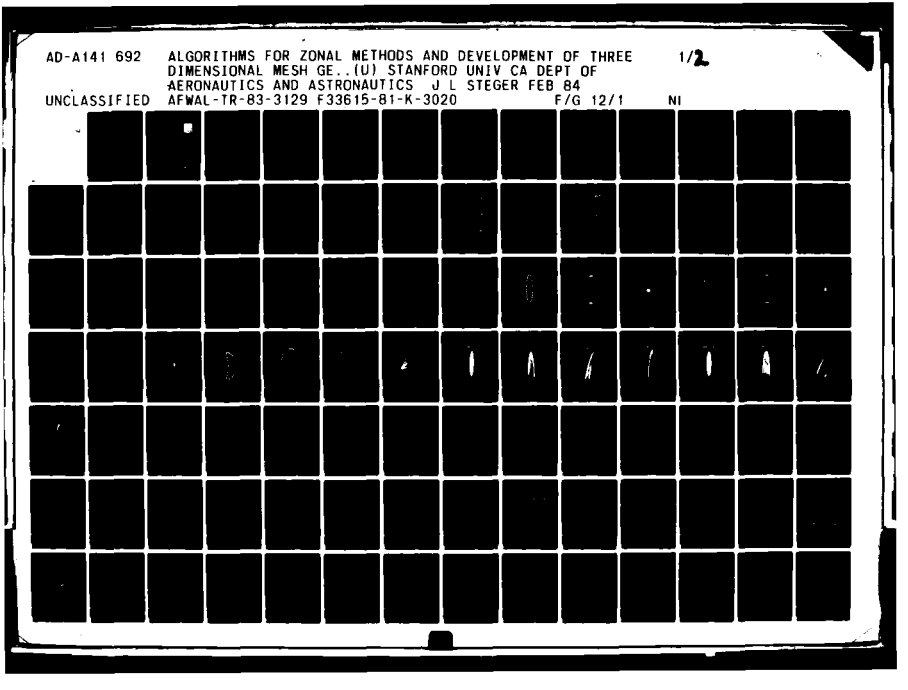
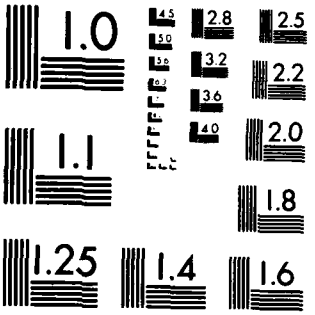
AD-A141 692 ALGORITHMS FOR ZONAL METHODS AND DEVELOPMENT OF THREE DIMENSIONAL MESH GE..(U) STANFORD UNIV CA DEPT OF AERONAUTICS AND ASTRONAUTICS J L STEGER FEB 84 1/2

UNCLASSIFIED AFWAL-TR-83-3129 F33615-81-K-3020 F/G 12/1 NI





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

(12)

AD-A141 692

AFWAL-TR-83-3129

ALGORITHMS FOR ZONAL METHODS AND DEVELOPMENT OF THREE DIMENSIONAL MESH GENERATION PROCEDURES

Joseph L. Steger
Department of Aeronautics and Astronautics
Stanford University
Stanford, California 94305

February 1984
Report for Period September 1980 to August 1983

DTIC FILE COPY

Approved for public release; distribution unlimited

DTIC ELECTE JUN 1 1984 S D A

FLIGHT DYNAMICS LABORATORY
AIR FORCE WRIGHT AERONAUTICAL LABORATORIES
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433

84 05 30 099

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture use, or sell any patented invention that may in any way be related thereto.

This report has been reviewed by the Office of Public Affairs (ASD/PA) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.

WILBUR L. HANKEY
Project Engineer

TOMMY J. KENT, Maj, USAF
Chief, Aerodynamics & Airframe Branch
Aeromechanics Division

FOR THE COMMANDER

RALPH W. HOLM, Col, USAF
Chief, Aeromechanics Division

"If your address has changed, if you wish to be removed from our mailing list, or if the addressee is no longer employed by your organization please notify AFWAL/FIMM, *W-PAFB, OH 45433 to help us maintain a current mailing list".*

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; Distribution Unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) AFWAL-TR-83-3129	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Stanford University 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State and ZIP Code) Department of Aeronautics and Astronautics Stanford, CA 94305	7b. ADDRESS (City, State and ZIP Code)
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Flight Dynamics Laboratory 8b. OFFICE SYMBOL (If applicable) AFWAL/FIMM	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER Air Force F33615-81-K-3020
8c. ADDRESS (City, State and ZIP Code) Wright-Patterson AFB, OH 45433	10. SOURCE OF FUNDING NOS. PROGRAM ELEMENT NO.: 61102F; PROJECT NO.: 2307; TASK NO.: N6; WORK UNIT NO.: 06
11. TITLE (Include Security Classification) Algorithms for Zonal Methods and Development of Three-Dim Mesh Generation Procedures	
12. PERSONAL AUTHOR(S) Joseph L. Steger	
13a. TYPE OF REPORT Final 13b. TIME COVERED FROM Sep 80 TO Aug 83	14. DATE OF REPORT (Yr., Mo., Day) February 1984 15. PAGE COUNT 92
16. SUPPLEMENTARY NOTATION	
17. COSATI CODES FIELD / GROUP / SUB. GR.	18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number) Computational Fluid Dynamics Zonal Methods, Grid Generation

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

The goal of this research was to further develop zonal methods and ~~3D~~ three dimensional grid generation procedures for application of finite difference methods to solve complex aircraft configurations. For the task of three dimensional grid generation both elliptic and hyperbolic methods were developed. A chimera grid scheme, that is, the use of overset multiple grid systems, was also tested in two dimensions. In zonal methods several new algorithms were developed. These included combining transonic potential codes with thin layer Navier-Stokes equations, unsteady transonic potential, Euler and vector potential codes. This report summarizes the various numerical algorithms that were studied.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS ☐	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL Wilbur L. Hankey	22b. TELEPHONE NUMBER (Include Area Code) (513) 255-2455 22c. OFFICE SYMBOL AFWAL/FIMM

DD FORM 1473, 83 APR EDITION OF 1 JAN 73 IS OBSOLETE.

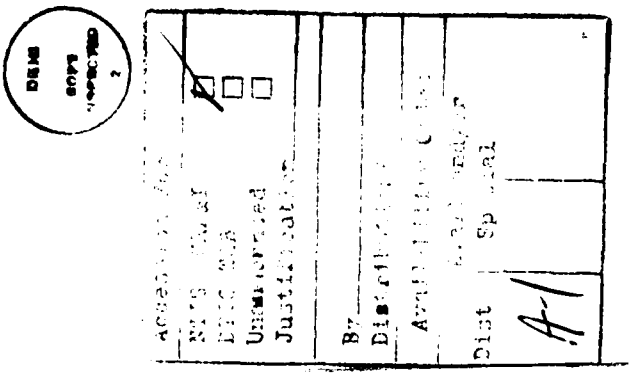
UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

PREFACE

The purpose of this research has been to further develop grid generation procedures and zonal methods so as to extend the applications of nonlinear finite difference methods to complex aircraft configurations. For the task of three dimensional grid generation both elliptic and hyperbolic grid generation methods were developed. A chimera grid scheme, that is, the use of overset multiple grid systems, was also tested in two dimensions. In our study of zonal methods several new algorithms and computer codes were developed. These included two and three dimensional zonal codes that match transonic potential equations with thin layer Navier-Stokes equations, an unsteady three dimensional transonic potential code, and a two dimensional zonal Euler and vector potential code.

This report summarizes the various numerical algorithms that were studied. Specific details of each algorithm are contained in a series of appendices that contain either a brief write-up or a copy of a published technical report. The brief report itself is broken into two main sections, Section I Grid Generation, and Section II Zonal Methods. In Section II an attempt is made to draw out some of the advantages and disadvantages of using a zonal method. This research contract also provided partial support to two graduate students in the Department of Aeronautics and Astronautics and resulted in the publication of several technical papers. Computer codes were also transferred to the Flight Dynamics Laboratory by Mr. Timothy Barth.



PRECEDING PAGE BLANK-NOT FILMED

TABLE OF CONTENTS

PRECEDING PAGE BLANK-NOT FILMED

SECTION I
GRID GENERATION

1. Background

In generating a grid about aircraft–like configurations several constraints should be kept in mind. To begin, the grids must be smoothly varying so as to maintain solution accuracy. The grids should also be body conforming to enhance solution accuracy, simplify the implementation of boundary conditions, and to minimize programming complexity. Finally, in order to use approximately factored implicit schemes and to maintain computational compatibility with vectorized machines, the grids should be well–ordered.

For a simple wing–body combination one can envision a single mapping procedure as illustrated in Figure (1). This warped spherical coordinate system is well–ordered, maps the body onto a single coordinate surface, and if properly generated, it can be sufficiently smooth. Moreover, the axis singularity is not a problem for Euler and thin layer Navier–Stokes equations that are transformed in general coordinates and strong conservation law form.

A single mapping such as that illustrated in Figure (1) is, of course, inadequate for complex aerodynamic configurations which can include engine nacelles, stores, etc. For these cases, subgrids, which are either embedded–to or overset–on the main wing–body conforming grid, are envisioned. Some possible grid configurations are illustrated in Figures (2) and (3) for two dimensional cross sections.

Embedding or oversetting meshes to account for complex configurations requires an immense effort in interfacing grids and numerical algorithm developments. Consequently, this university contract was restricted to the more fundamental task of generating only the main body–conforming grid using elliptic partial differential grid generation equations with clustering terms. We have, however, begun development of a three dimensional hyperbolic

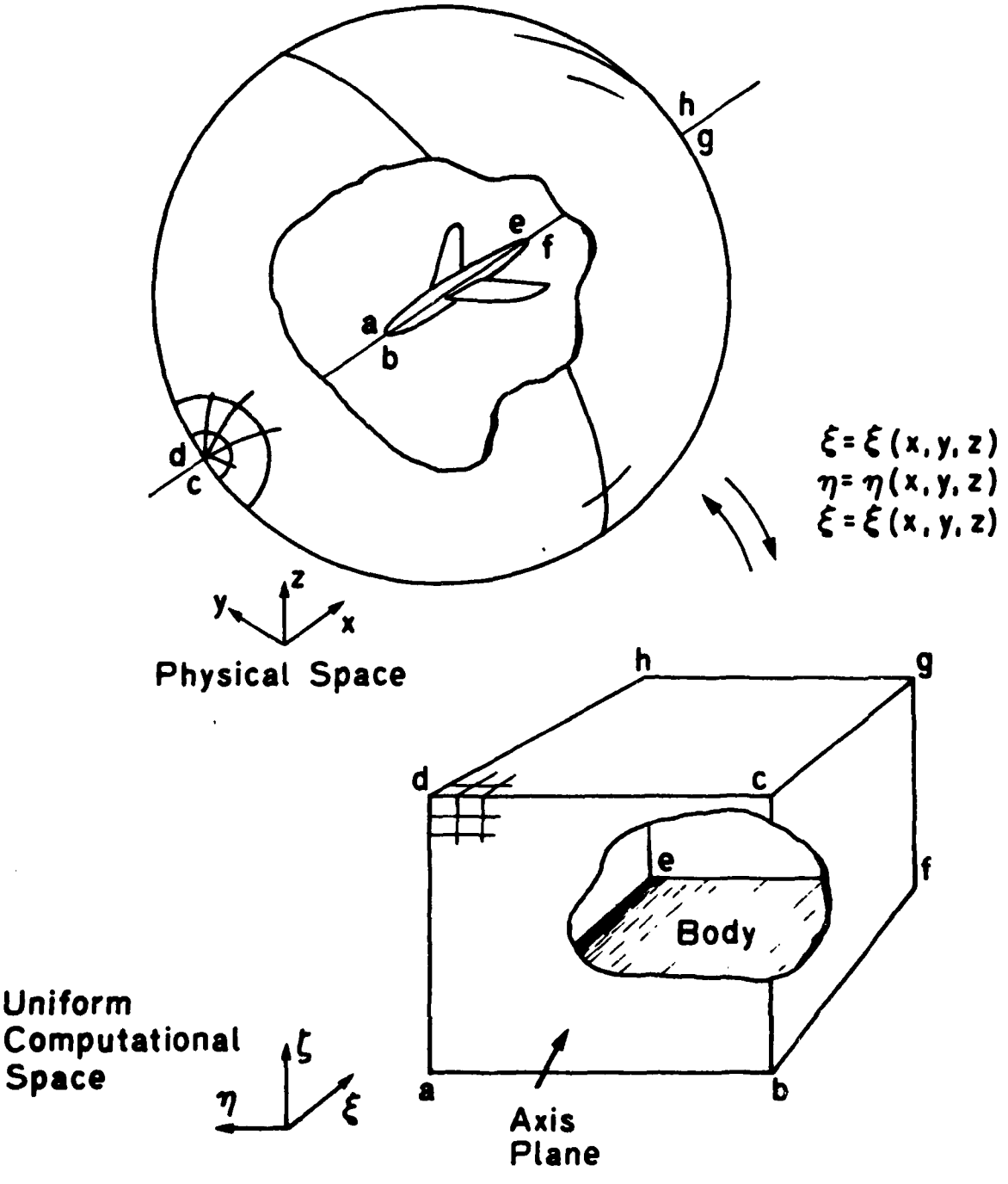


Figure 1. Well-Ordered Warped Spherical Grid Mapping

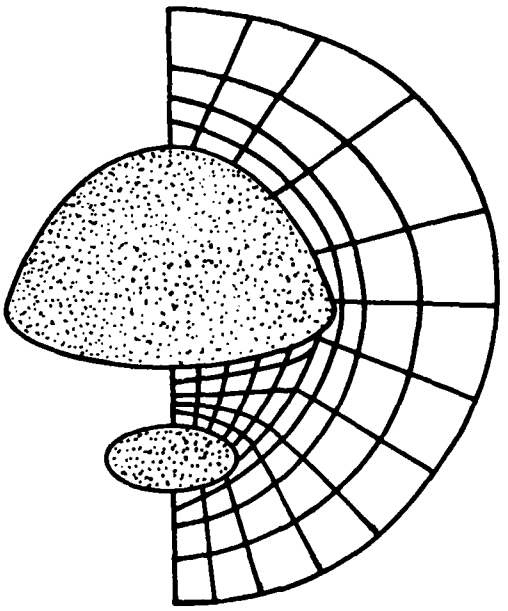


Figure 2. Example of Mesh Embedding in Two Dimensions

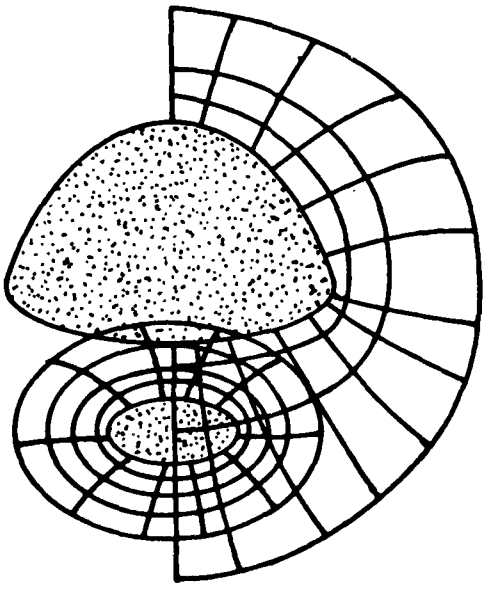


Figure 3. Example of Mesh Overlapping in Two Dimensions

grid generation procedure, and, in two dimensions, we have indeed studied overset grids.

2. Grid Generation

As noted earlier, we wish to build a single main grid that is smoothly continuous, maps the body onto a constant coordinate surface, and is well–ordered. The resulting mesh must also use surface as well as field grid points efficiently. Clustering of grid points near a body to resolve viscous terms should not cause unwanted extraneous points in the free stream. These considerations lead to the use of warped spherical and cylindrical coordinate systems.

To construct this main spherical grid we have developed grid generation codes using both elliptic and hyperbolic partial differential equations. The elliptic grid generation procedure has been written up in the form of a technical paper and was presented by Mr. Reese Sorenson at an ASME specialist meeting. This paper, which describes the method and shows results, is reproduced in Appendix A.

A paper describing our work in three dimensional hyperbolic grid generation procedure is in preparation. Details of this method are described in an abbreviated write–up as Appendix B.

In general the elliptic grid generator is more powerful as it can generate a smooth grid interior to user specified inner and outer boundaries. The hyperbolic method generates a grid exterior to a user specified inner boundary. The outer boundary location is not known in advance (although one could iterate on its location, but not its placement of points). However, the hyperbolic solver can generate an orthogonal grid and it is a much more efficient code to use than the elliptic solver. For cases in which it is suited, it also requires less user input. The elliptic solver, however, is more reliable.

3. Overset Grids

A paper describing a chimera or overset grid scheme in two dimensions is attached as

Appendix C. This paper was presented at the same ASME meeting as the three dimensional elliptic solver. The various results, which are obtained with linear incompressible flow equations, verify the feasibility of this approach, but considerably more research is needed.

SECTION II
ZONAL METHODS

1. Background

Just as one wants to limit the number of grid points to only those action regions in which they are needed, one would also like to limit the complexity of the governing partial differential equations to appropriate action zones. The Navier–Stokes equations, for example, are not needed to describe the flow in the inviscid far field. Indeed, even nonlinear full potential theory is not needed there, as a linear potential approximation is quite adequate.

In principle, considerable computational work can be saved by using just that simplified governing equation set that suffices for a given region of the flow field. Computer storage can also be reduced. However, such a zonal method is not without pitfalls. For one, programming and matching together several numerical methods increases the overall complexity of the computer code and its data base. This is especially undersirable on parallel processors. Moreover, unless one is very careful in matching the different numerical algorithms and governing equations, overall stability and iterative convergence can be impaired. Numerical stability could conceivably decline so much that the simplified zonal scheme could be more inefficient than a straight Navier–Stokes algorithm with a well stretched grid.

Such negative arguments may be offset by additional advantages for zonal schemes. For one, steady state numerical algorithms for potential flow are far advanced over those for Euler or Navier–Stokes equations. This is because the simpler scalar potential equation is much easier to optimize for steady state convergence. A well matched zonal method may therefore have better convergence than, say, a Navier–Stokes scheme alone, simply because the outer flow region can be converged at a much faster rate. Another, albeit intangible aspect of the zonal method is that one is forced to keep the numerical algorithms

compatible, and if the code is properly designed, either algorithm could be readily used in a stand alone mode.

Arguments for and against zonal methods will be summarized later. Clearly, a zonal scheme has some reduced computer storage over a Navier–Stokes alone algorithm, and it may have other significant computational benefits. Consequently, we have coded and tested a zonal algorithm. In our tests we have essentially combined a transonic full potential code with an Euler or thin layer Navier–Stokes code. The zonal code was tested in first two and then three dimensions.

2. The Zonal Codes

The three dimensional zonal code which we have written matches an unsteady conservative full potential code with an Euler or thin layer Navier–Stokes code. Details of the full potential code, partially developed for this application, were presented as an AIAA paper attached as Appendix D. The zonal code itself has not been published, but Mr. Jack Striegberger is using the code in his Ph.D. thesis project, so it will ultimately be fully documented. Mr. Timothy Barth has transferred the code to AFFDL.

A two dimensional zonal code has also been written which combines a flux vector split Euler code with a potential and vector potential code. Using this dual potential combination, the outer flow is able to correctly convect entropy. This has lead to a very versatile zonal code in which we are able to match the equation zones in a much more elegant (and simpler) way. Details of this method have also been presented at an AIAA technical meeting, and this paper is presented as Appendix E.

3. Reflections

The zonal codes that we have written work and save both computer time and some storage. They are, however, more complex than a single Euler or Navier–Stokes code. Moreover, the zonal codes are not as readily generalized to a new problem because somewhere in the flow field they use simplifying assumptions. In fact, a zonal code makes a

trade between computer time and engineering time. A zonal code that gives the same result as a Navier–Stokes code will be cheaper to run on the computer, but it will require more engineering development time. So the zonal code is ideal for optimizing a given configuration because, once the code is set up, it will run more efficiently. If one is continually changing the layout of the configuration and the type of flow field being solved, then a nonzonal general code may be more economical because the engineering time for the first solution will be less.

The area in which more research is needed with zonal codes is in how to interface the zones more tightly with a minimum of bookkeeping. The two dimensional zonal code described in Appendix E presents one approach that I believe we can and should generalize.

In the zonal method described in Appendix E we solve (using $u = \phi_x + \psi_y, v = \phi_y - \psi_x$)

$$(\rho u)_x + (\rho v)_y = 0 \tag{1a}$$

$$v_x - u_y = \omega \tag{1b}$$

$$us_x + vs_y = 0 \tag{1c}$$

$$\rho = \rho(u^2 + v^2, s) \tag{1d}$$

throughout the entire flow. These equations can convect but not produce entropy. In an entropy producing zone (e.g., around a shock, see Figure (1b) of Appendix E) the inviscid conservation law equations of mass, momentum, and energy are solved. From this solution we obtain the value of ω that feeds into Equation (1b) by forming $\omega = (u_y - v_x)$ directly from the conservation law solutions. Elsewhere ω is evaluated from

$$-\omega = (\gamma M^2)^{-1}(vs_x - us_y)$$

so that Equation (1b) is the Crocco equation. Likewise in the shock zone s is overloaded directly from the conservation laws in place of Equation (1c).

The point of this is that Equations (1) are used throughout and so there is less logic to code by avoiding a zone boundary (especially so since we in fact use ϕ and ψ as variables). Moreover, because an implicit solver is used, information is spread throughout without any lagging of zone boundaries. Now, in fact, it would take fewer operations per iteration if Equations (1) were turned off in the conservation law zone. But the code would, as just stated above, be more complex and likely require more iterations between zones to reach a steady state if this were done.

What we have done in this zonal method is to use a single "simple" equation set throughout. In "complex" flow zones, a more complete set of equations is used, but their effect is imposed by means of a right hand side forcing function, not by means of a zonal boundary. This, I believe, makes this new zonal approach much more versatile. For example, in viscous flow we should be able to evaluate the ω of Equation (1b) from a Navier–Stokes or boundary layer equation zone. Ideally we could use Equation Set (1) everywhere and use more complicated equations only as a means to overwrite ω and s. Such an approach is currently being formulated.

To conclude, my feeling is that zonal codes are needed and have their place, generally in a large engineering design environment. Small engineering teams that have to model a variety of different flow fields and configurations should use more general codes that are easier to set up for a given problem. In designing new zonal codes it is important that we try to keep the number of equation sets and zone interface boundaries to a minimum for simplicity. The type of zonal approach developed in Appendix E is one such attempt, and it appears to be very promising.

SECTION III
CONCLUSIONS

Grid generation procedures and zonal solution methods were studied. Both elliptic and hyperbolic three dimensional grid generation procedures were developed. The hyperbolic grid generator is especially easy to use and it is very efficient. It can fail, however, whenever the body surface is discontinuous or the used specified surface grid distribution is too irregular. The elliptic solver is more robust, but it requires much more user input and much more computer time.

From our experience with zonal codes, we concluded that they have their place, particularly for design applications where maximum computational efficiency is required. However, a major effort must be made to reduce the complexity of zonal codes. An approach which interfaces the zones through forcing functions rather than boundary conditions was developed, and it appears to offer this possibility.

APPENDIX A

GRID GENERATION IN THREE DIMENSIONS BY POISSON EQUATIONS WITH CONTROL OF CELL SIZE AND SKEWNESS AT BOUNDARY SURFACES

Mini-Symposium on Advances in Grid Generation

ASME Fluids Engineering Conference

June 20-22, 1983, Houston, Texas

by

Reese L. Sorenson
Research Scientist
NASA Ames Research Center
Moffett Field, California

and

Joseph L. Steger
Associate Professor
Stanford University
Stanford, California

ABSTRACT

An algorithm for generating computational grids about arbitrary three-dimensional bodies is developed. The elliptic partial differential equation (PDE) approach developed by Steger and Sorenson and used in the NASA computer program GRAPE is extended from two to three dimensions. Forcing functions which are found automatically by the algorithm give the user the ability to control mesh cell size and skewness at boundary surfaces. This algorithm, as is typical of PDE grid generators, gives smooth grid lines and spacing in the interior of the grid. The method is applied to a rectilinear wind-tunnel case and to two body shapes in spherical coordinates.

NOMENCLATURE

a	coefficient in forcing function influencing decay rate of control at boundary
b	coefficient in forcing function influencing decay rate of control at boundary
c	coefficient in forcing function influencing decay rate of control at boundary
J	Jacobian of transformation, determinant of M
M	matrix of transformation metrics
P	forcing function in Poisson equation
$\tilde{P}$	factor in P, giving control of cell size and skewness at boundary
Q	forcing function in Poisson equation
$\tilde{Q}$	factor in Q, giving control of cell size and skewness at boundary
R	forcing function in Poisson equation
$\tilde{R}$	factor in R, giving control of cell size and skewness at boundary
$\vec{r}$	column vector having elements x,y,z
$\overrightarrow{RHS}$	term used in Eq. (7) in solving for P,Q,R
S	distance along line of increasing ζ in real domain
T	superscript indicating transpose of a matrix
x	independent variable in real domain, Cartesian coordinate
y	independent variable in real domain, Cartesian coordinate
z	independent variable in real domain, Cartesian coordinate
α	nonlinear coefficient in Eq. (2a)
γ	signed *cofactor* of the matrix M
Γ	matrix having elements γ
Δ	finite difference
ζ	independent variable in computational domain
η	independent variable in computational domain
θ	angle down from axis toward equator in spherical coordinates
ξ	independent variable in computational domain
ρ	distance from origin in spherical coordinates
ϕ	angle around axis in spherical coordinates
ω	relaxation parameter for point-SOR
i	row number in matrices M and Γ
j	column number in matrices M and Γ

INTRODUCTION

The ability to generate grids about arbitrary three-dimensional aerodynamic configurations stands today as one of the critical pacing items in computational fluid dynamics (1). Of the various methods for generating grids, the elliptic partial differential equation technique (2-6), with its inherent smoothness, has proven to be one of the most automatic and general approaches. In two-dimensional applications this has been especially true when the elliptic grid-generation equations are combined with an algorithm that automatically chooses inhomogeneous terms to give the user control of mesh cell size and skewness at boundaries. Such an algorithm, as developed in Refs. 7 and 8, has resulted in the

widely used NASA computer program GRAPE[1] (8). The extension of the GRAPE algorithm to three dimensions has been long overdue and is the subject of this paper.

In the algorithm developed here, three-dimensional elliptic partial differential grid-generation equations are modified to give the user control of the spacing and skewness of mesh lines that approach the boundary. This is accomplished by adding to the grid generation equations a set of forcing terms which are automatically evaluated by imposing differential equation constraints of arc length control and surface orthogonality at the boundary. The user has only to input the desired grid spacing at the boundary surface. A description of these grid-generation equations and a numerical solution procedure are developed in the main part of this paper. Because a spherical topology is one of the most efficient grid systems in three dimensions, the grid-generation algorithm is further modified to avoid difficulties in generating grids near a spherical or cylindrical axis singularity. These details and grid results in both rectangular and spherical grid topologies are presented in the remainder of the paper.

GRID-GENERATION EQUATIONS

The elliptic partial differential equation grid-generation approach has been extensively developed elsewhere (9). Elements of this approach are reviewed below to develop source terms that automatically enforce interior mesh line orthogonality to a boundary surface and give the user control of the step size between the boundary and the next interior surface.

It is required that the mapping between physical space x,y,z and computational space ξ,η,ζ satisfy the Poisson equations

$$\xi_{xx} + \xi_{yy} + \xi_{zz} = P(\xi,\eta,\zeta) \tag{1a}$$

$$\eta_{xx} + \eta_{yy} + \eta_{zz} = Q(\xi,\eta,\zeta) \tag{1b}$$

$$\zeta_{xx} + \zeta_{yy} + \zeta_{zz} = R(\xi,\eta,\zeta) \tag{1c}$$

Given proper choice of the source terms P,Q,R, these equations satisfy the maximum principle and thus ensure a one-to-one mapping. Equation (1) is conveniently solved numerically in the uniform computational space, ξ,η,ζ. The equations, so transformed (10-13), are

$$\alpha_{11}\vec{r}_{\xi\xi} + \alpha_{22}\vec{r}_{\eta\eta} + \alpha_{33}\vec{r}_{\zeta\zeta} + 2(\alpha_{12}\vec{r}_{\xi\eta} + \alpha_{13}\vec{r}_{\xi\zeta} + \alpha_{23}\vec{r}_{\eta\zeta}) = -J^2(P\vec{r}_{\xi} + Q\vec{r}_{\eta} + R\vec{r}_{\zeta}) \tag{2a}$$

where

$$\vec{r} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \qquad \alpha_{ij} = \sum_{m=1}^{3} \gamma_{mi}\gamma_{mj}, \tag{2b,2c}$$

γ_{ij} is the ijth signed cofactor of the matrix M

$$M = \begin{bmatrix} x_\xi & x_\eta & x_\zeta \\ y_\xi & y_\eta & y_\zeta \\ z_\xi & z_\eta & z_\zeta \end{bmatrix} \tag{2d}$$

and the Jacobian J is the determinant of M.

[1]An acronym derived from "GRids about Airfoils using Poisson's Equation."

In the present application the inhomogeneous terms P,Q,R control grid spacing and skewness for mesh cells adjacent to the body boundary surface, $\zeta = 0$. The forcing terms are chosen to be

$$P(\xi,\eta,\zeta) = \tilde{P}(\xi,\eta)e^{-a\zeta} \tag{3a}$$

$$Q(\xi,\eta,\zeta) = \tilde{Q}(\xi,\eta)e^{-b\zeta} \tag{3b}$$

$$R(\xi,\eta,\zeta) = \tilde{R}(\xi,\eta)e^{-c\zeta} \tag{3c}$$

when the exponential factors cause the control to dissipate or relax with distance from the boundary $\zeta = 0$. Relaxing the control with distance from the $\zeta = 0$ boundary is necessary so as not to overly constrain the grid lines with respect to the opposing ($\zeta = \zeta_{max}$) boundary. The positive constants a,b,c influence the rate of decay of the boundary control.

In most elliptic grid-generation techniques, the points on boundaries are user-specified. Thus, the ξ,η distribution of grid points is specified on the $\zeta = 0$ surface. Figure 1 shows a typical mesh cell touching a

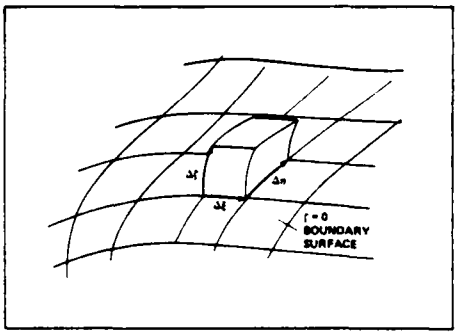


Fig. 1 Typical mesh cell touching $\zeta = 0$ surface

$\zeta = 0$ surface, with ξ and η varying over that surface. The condition for orthogonality of the grid lines intersecting the $\zeta = 0$ boundary surface is that the unit vectors in the ξ and ζ directions and in the η and ζ directions be mutually normal. These two conditions can be expressed by the vector dot products

$$\vec{r}_{\xi} \cdot \vec{r}_{\zeta} = 0 \tag{4a}$$

$$\vec{r}_{\eta} \cdot \vec{r}_{\zeta} = 0 \tag{4b}$$

To control the cell size on the $\zeta = 0$ boundary, the "height" of the cells must be regulated to some prespecified value. Letting S be that height, the distance along a line of increasing ζ, we wish to specify $\Delta S/\Delta\zeta$ at the $\zeta = 0$ boundary. In differential form this third boundary-control equation can be expressed as

$$\vec{r}_{\zeta} \cdot \vec{r}_{\zeta} = \left(\frac{\partial S}{\partial \zeta}\right)^2 \tag{4c}$$

The solution procedure described below requires that Eq. (4) be solved for the derivatives with respect to ζ at the surface, giving

$$z_\zeta = \frac{\partial S/\partial\zeta}{(\gamma_{13}/\gamma_{33} + \gamma_{23}/\gamma_{33} + 1)^{1/2}} \tag{5a}$$

$$x_\zeta = \frac{\gamma_{13}z_\zeta}{\gamma_{33}} \tag{5b}$$

$$y_\zeta = \frac{\gamma_{23} z_\zeta}{\gamma_{33}} \tag{5c}$$

The surface control equations expressed by either Eqs. (4) or (5) are solved simultaneously with the interior grid-generation equations given by Eq. (2). Because x,y,z are specified on the boundary, the surface control equations supply three additional relations that can be used to determine the unknown values of P,Q,R. An iterative solution process is used as described below.

SOLUTION PROCEDURE

An iterative procedure is used to solve the grid-generation equations. Each iteration is in two distinct parts. In the first part we begin with the x,y,z from the initial conditions or the previous iteration, and update values of the P,Q,R terms. The second part of the iteration step updates values of x,y,z at each point in the field using the new values for P,Q,R.

To update the P,Q,R, we first note that solving Eq. (2) produces a grid for an appropriate choice of P,Q,R. We wish to impose constraints on grid cells at the boundary $\zeta = 0$, and thus determine P,Q,R. Equation (4) gives the constraints and Eq. (5) gives the same constraints expressed as requirements on derivatives of $\vec{r}$ with respect to ζ at the boundary. The derivatives in Eq. (5), along with difference approximations for all other first and second partial derivatives of $\vec{r}$ with respect to ξ,η,ζ, are substituted into Eq. (2), which is solved for P,Q,R.

To obtain the difference approximations for first and second partial derivitives of $\vec{r}$ with respect to ξ,η,ζ on the $\zeta = 0$ boundary surface, we proceed as follows. Because x,y,z are specified at $\zeta = 0$, first and second partial derivatives of $\vec{r}$ with respect to ξ and η on that surface can be found by differencing fixed boundary points. Those derivatives, combined with user specification of $\partial S/\partial\zeta$, are used in Eq. (5) to determine derivatives $\vec{r}_\zeta$. Second partial derivatives $\vec{r}_{\xi\zeta}$ and $\vec{r}_{\eta\zeta}$ are found by differencing $\vec{r}_\zeta$ with respect to ξ and η. Thus, the only derivatives lacking in Eq. (2) are $\vec{r}_{\eta\eta}$. These are found by differencing the solution for $\vec{r}$ at and near the surface using the current interior grid solution.

Thus, at each $\zeta = 0$ point, Eq. (2a) are three equations which can be solved for the three unknowns P,Q,R. From Eq. (3) it can be seen that at $\zeta = 0$, $P(\xi,\eta,\zeta)$ reduces to $P(\xi,\eta)$, and similarly for Q and R. From Eq. (2) $\overrightarrow{RHS}$ is defined as

$$\overrightarrow{RHS} = -J^{-2}[a_{11}\vec{r}_{\xi\xi} + a_{22}\vec{r}_{\eta\eta} + a_{33}\vec{r}_{\zeta\zeta} + 2(a_{12}\vec{r}_{\xi\eta} + a_{13}\vec{r}_{\xi\zeta} + a_{23}\vec{r}_{\eta\zeta})] \tag{6}$$

The solution then is

$$\begin{bmatrix} \tilde{P} \\ \tilde{Q} \\ \tilde{R} \end{bmatrix} = M^{-1}\,\overrightarrow{RHS} = \Gamma^T\,\overrightarrow{RHS}/J \tag{7}$$

where Γ is the matrix having elements γ_{ij}. For the whole field P,Q,R are then found by multiplying $\tilde{P},\tilde{Q},\tilde{R}$ by the appropriate exponential factors as in Eq. (3).

The second part of each iteration step is to use the new values for P,Q,R in Eq. (2) to find new x,y,z everywhere in the field. In this research effort the iterative solution procedure, chosen for ease of coding, was point-SOR. Thus, the P,Q,R terms necessary to cause the grid to have the desired behavior at the boundary are found automatically along with the x,y,z in the interior as this two-part iteration scheme proceeds to convergence.

SPHERICAL GRIDS

From a computational point of view a spherical topology is one of the most efficient grid systems for three-dimensional bodies because a spherical grid saves points in the far field. For example, in Navier-Stokes calculations that use highly clustered grids near the body boundary with a single rectilinear coordinate system, the fine grid near the body can extend into the far field. This does not occur with spherical coordinates. A spherical coordinate system does introduce an axis singularity, but experience shows that an axis singularity can be readily handled in flow codes that use general ξ,η,ζ coordinates and solve the flow equations in conservative form (14).

For grid generation using Eq. (2), the axis singularity of the spherical coordinate system has proven to be difficult. Following a suggestion of J. K. Hodge (private communication, 1977), M. Vinokur and J. L. Steger (private communication, 1978) found that one way to avoid the axis singularity was to "convert" Eq. (1) into a pseudospherical equation. Instead of using true spherical Poisson equations, the independent variables x,y,z are simply replaced with ρ,θ,ϕ in Eq. (2) for spherical coordinates or by ρ,ϕ,z for cylindrical coordinates. See Fig. 2. Thus, Eq. (1) is replaced by

$$\xi_{\rho\rho} + \xi_{\theta\theta} + \xi_{\phi\phi} = P(\xi,\eta,\zeta) \tag{8a}$$

$$\eta_{\rho\rho} + \eta_{\theta\theta} + \eta_{\phi\phi} = Q(\xi,\eta,\zeta) \tag{8b}$$

$$\zeta_{\rho\rho} + \zeta_{\theta\theta} + \zeta_{\phi\phi} = R(\xi,\eta,\zeta) \tag{8c}$$

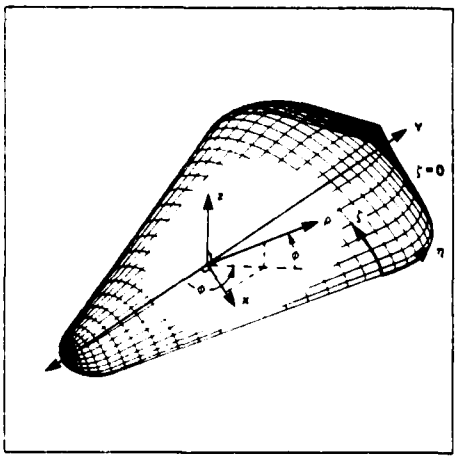


Fig. 2 Cut away sketch showing computational variables and spherical coordinates

Equation (8) is not in the form of a true Laplacian operator in spherical coordinates. It is, however, an elliptic equation that satisfies the maximum principle and it generates a smoothly varying grid just as Eq. (1) does. At the axis, ϕ varies monotonically with ξ (see Fig. 2) and no singular behavior is encountered. Transformation of Eq. (8) to uniform computational space results in Eq. (2) with $\vec{r}$ replaced by

$$\begin{bmatrix} P \\ \vdots \\ \vdots \end{bmatrix}.$$

Solution of the spherical variable form of Eq. (2) proceeds much as before. Boundary-point values of x,y,z are specified along with an initial guess of the interior-point values of x,y,z. At each of the points, values of x,y,z are then converted in the usual way to spherical values of ρ,θ,ϕ. The grid-generation equations are then solved by relaxation, and once a solution is obtained, it is converted back to x,y,z values at each grid point. Equation (8) does not generate the same solution as Eq. (1), but the solution is one that can be just as satisfactory. It must be cautioned that if ϕ varies from 0 to 2π in a periodic grid, then one must difference in a manner which accounts for a discontinuity of the function, but not of the derivative by adding or subtracting 2π where appropriate.

The surface clustering and orthogonality relations as developed previously are written in terms of x,y,z. Rather than attempt to rework these relations in spherical variables, we simply convert ρ,θ,ϕ variables back to x,y,z variables to enforce Eq. (4). That is, whenever P,Q,R must be evaluated, grid-point values of ρ,θ,ϕ are converted back to x,y,z. Values of P,Q,R are found, and then x,y,z values are converted back to ρ,θ,ϕ. Since this transformation and its inverse need be done only for the first few (typically 3) shells of points near the body surface, the increase in computation is not significant.

RESULTS

The method has been coded to control surface orthogonality and step-size spacing only at the inner ($\zeta = 0$) boundary. The same 3-D computer code can be used to generate grids either for rectangular grid topologies using Eq. (1) in terms of x,y,z coordinates, or warped, spherical grid topologies using Eq. (8) for ρ,θ,ϕ coordinates.

Rectilinear Grids

Rectilinear three-dimensional grids with automatic surface step-size control have been generated. A rectangular wind tunnel with a bump on the floor is used as a test case. Since this case fits within the rectilinear topology, Eq. (1) is used as the grid generation equation. Figure 3(a) illustrates the floor ($\zeta = 0$) boundary surface. The grid in this case has 40 points in the ξ or streamwise direction, 20 points in the η or spanwise direction, and 25 points in the ζ or vertical direction. The mesh lines intersecting the floor are required to do so normally, and a spacing normal to the floor equal to 0.0025 times the height of the tunnel is enforced. That spacing is approximately 1/20th of what would result if the points were equally spaced in the vertical direction.

Figure 3(b) shows the fifth ζ = constant coordinate surface above the floor. The control of point spacing normal to the $\zeta = 0$ boundary surface is shown here by the uniform proximity of this surface to the floor. Figure 3(c) shows the fifteenth ζ = constant surface, above the floor. Here, the bump has begun to "flatten" since the grid has a flat upper-boundary plane. An example of a different family of coordinate surfaces, a ξ = constant surface is shown in Fig. 3(d). It rides the crest of the bump from side to side in the tunnel. Figure 3(e) shows an example of the third family of coordinate surfaces, an η = constant surface. It is the fifth such surface, counting from the near side of the tunnel. The viewpoint here is normal to that surface. Figure 3(f) shows a closeup of a region of the surface shown on the previous Fig. 3(e). It is the region at the right-hand side of the base of the bump. Note that the spacing normal to the bottom boundary is

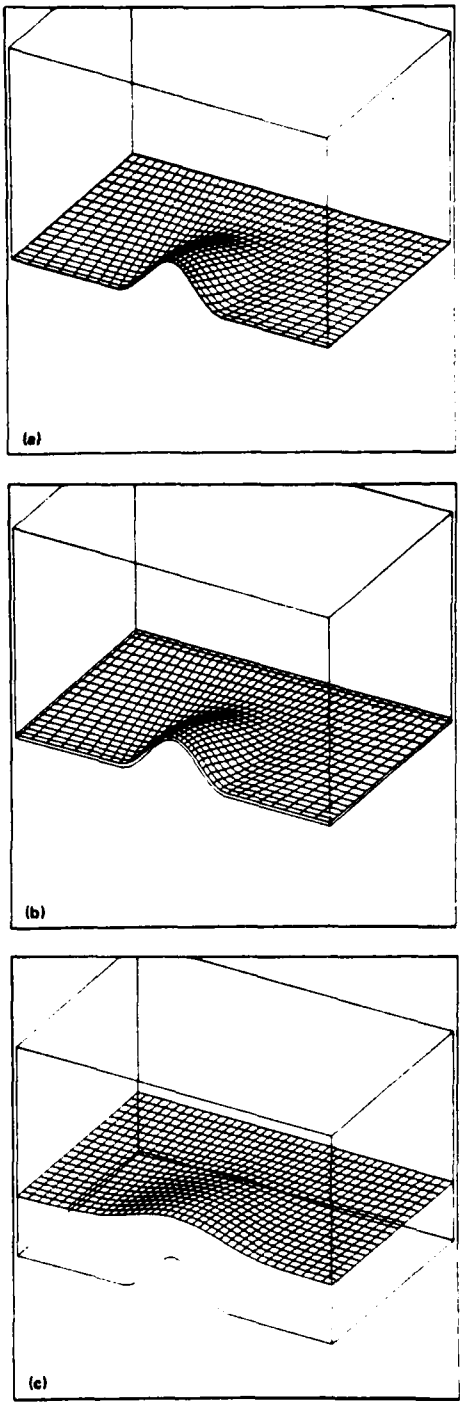
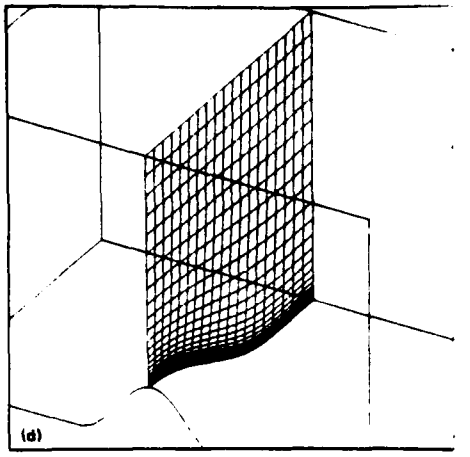
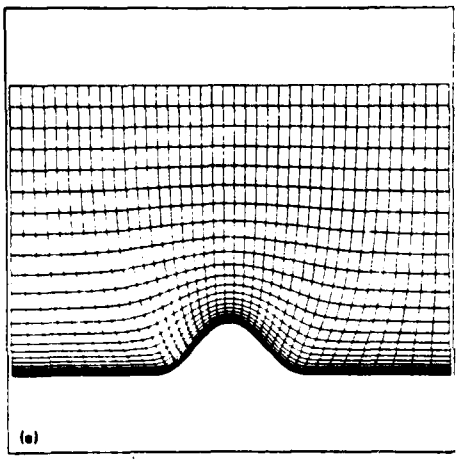


Fig. 3 Grid for rectangular wind tunnel with bump on floor
(a) Inner ($\zeta = 0$) boundary surface (the floor), showing bump
(b) Fifth ζ = constant coordinate surface above the floor
(c) Fifteenth ζ = constant coordinate surface above the floor





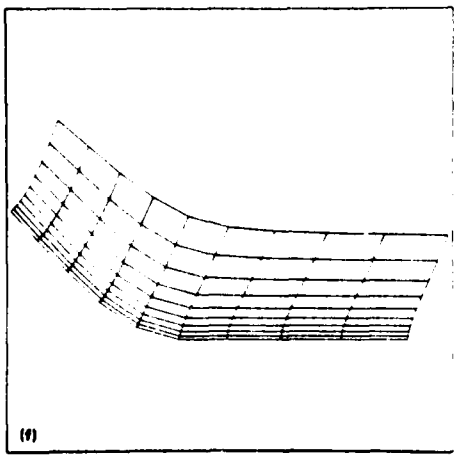


Fig. 3 Concluded
(d) ξ = constant coordinate surface on top of bump
(e) Fifth η = constant surface passing over top of the bump, orthogonal view
(f) Closeup of region near base of bump

constant and very small. That spacing would tend to be much larger and nonuniform in this concave region if the grid had been generated without the forcing terms.

Warped Spherical Grids

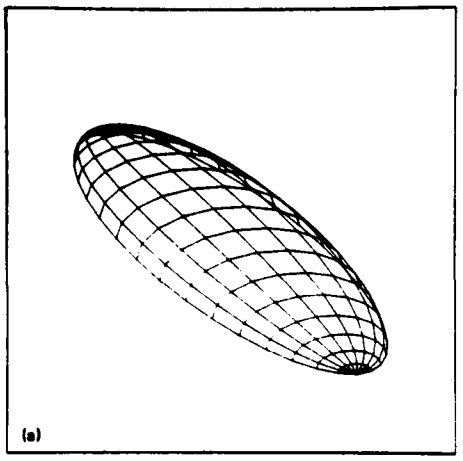
Grids for fuselage shapes and wing shapes have been generated using the spherical coordinate system and solving Eq. (8). As of this writing some convergence problems are encountered with fine grids using spherical topology. This is currently under investigation. The results presented here for spherical topology are limited to moderately coarse grids.

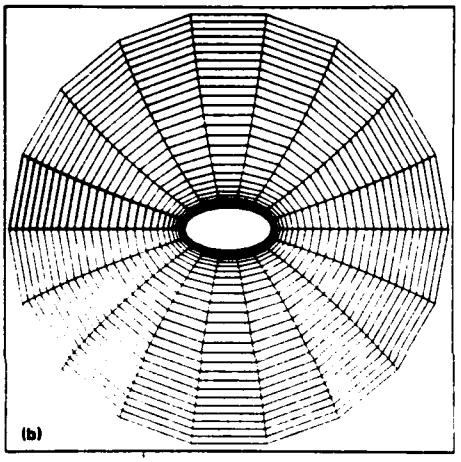
A grid having 20 points in the circumferential (ξ) direction, 19 points in the pole-to-pole (η) direction, and 30 points in the outward (ζ) direction has been generated about a 10 by 2 by 1 ellipsoid. Figure 4(a) illustrates the specified ξ and η distributions about the top half of this ellipsoidal body. Figure 4(b) shows the η = constant coordinate surface about the "equator," while Fig. 4(c) shows a closeup of the 10 innermost lines in the equatorial plane. Note that the spacing normal to the body is controlled. Figure 4(d) shows a different η = constant coordinate surface, this one nearer to the pole than the equator. It is not a plane, but rather more in the shape of a bowl, open toward the lower right. An example of a different family of coordinate surfaces, a ξ = constant surface, is shown in Fig. 4(e), extending from pole to pole. Figure 4(f) is a closeup of the region of Fig. 4(e) near the left-hand pole and near the body. The results indicate that the surface orthogonality control and the clustering control are working well, and that the solution about the axis is well behaved. Examples of the third kind of coordinate surface, ζ = constant, are visually indistinguishable from the body, Fig. 4(a).

On an axis, θ is either 0 or π and values of ϕ and ρ are obtained by parabolic extrapolation to the axis from the two nearest points in the η direction. The three conditions used to determine the parabola are that the mesh line must pass through the two nearest points in the η direction, and that it must have zero slope relative to the axis as it crosses the axis. Values of ρ for each point on the axis could be found by extrapolating along any of the lines encountered as ξ varies from 0 to ξ_{max}, and in fact the ρ used is an average of all those values.

Some views of another spherical grid, this one having 30 by 19 by 30 points, are shown in Fig. 5. This grid is about a wing having the same 5 to 1 ratio of span to midchord elliptical planform as the ellipsoid above, and a 19% thick airfoil section. The only adaptation necessary here is that the $\hat{P},\hat{Q},\hat{R}$ terms at the trailing edge are replaced by the average of their values immediately above and below the edge, i.e., they are "averaged" across the edge in the ξ direction. The airfoil section and the innermost four grid lines are shown in Fig. 5(a). The normal spacing and the angularity of the lines intersecting the body are nicely controlled. The sharp trailing edge is created successfully, as shown in Fig. 5(b). Figure 5(c) shows the far-field behind the trailing edge, and illustrates the ability of the PDE method to generate a smooth grid over a sharp corner.

In all of the above cases, values for the relaxation parameter ω in the point-SOR varied from 0.8 to 1.6. Values of the a,b,c parameters in Eq. (3) were approximately 0.5. The number of iterations necessary varied from 25 to 150.





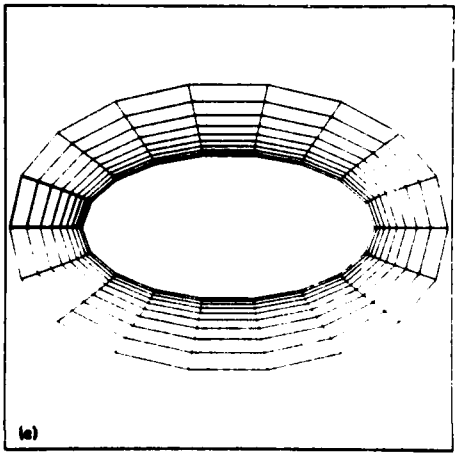
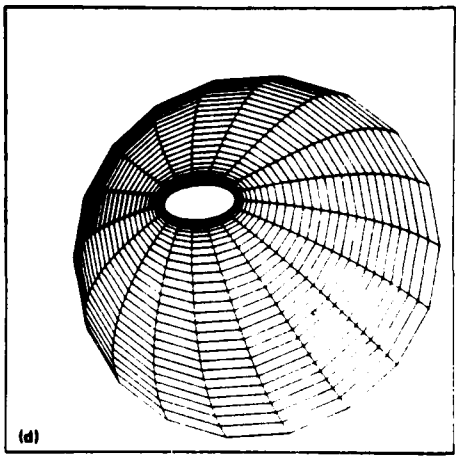
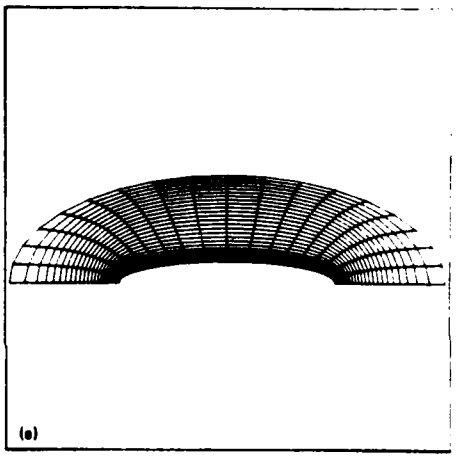


Fig. 4 Grid about ellipsoid using spherical coordinates
(a) Top half of ellipsoidal body surface
(b) Tenth η = constant coordinate surface, i.e., equatorial plane
(c) Closeup of innermost 10 circumferential lines in equatorial plane





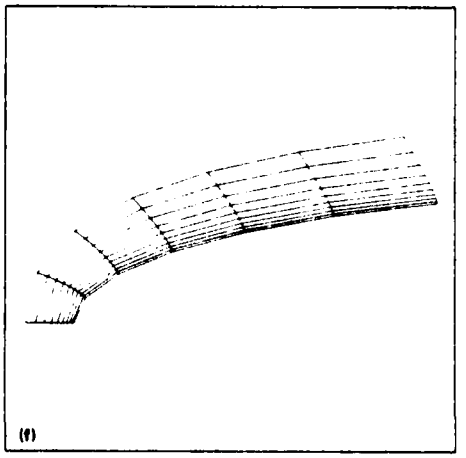
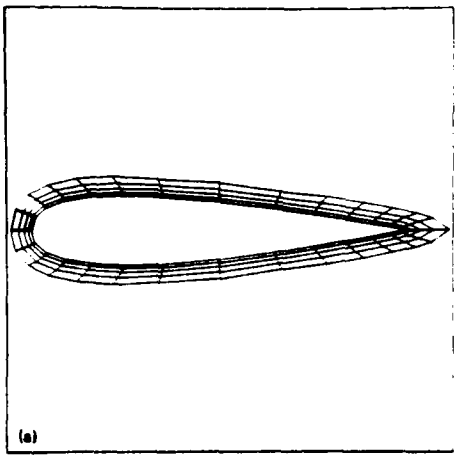
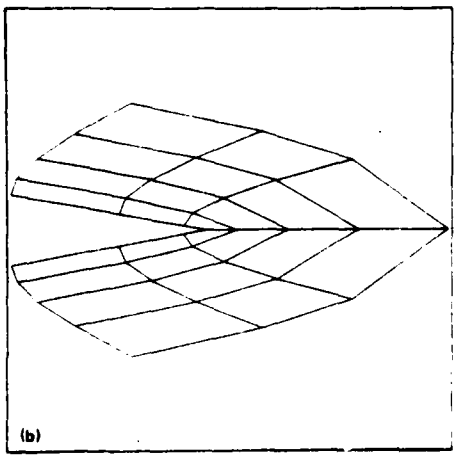


Fig. 4 Concluded
(d) Fifth bowl (η = constant coordinate surface), counting toward equator from pole
(e) ξ = constant coordinate surface extending from pole to pole
(f) Closeup of ξ = constant coordinate surface, near pole and near body





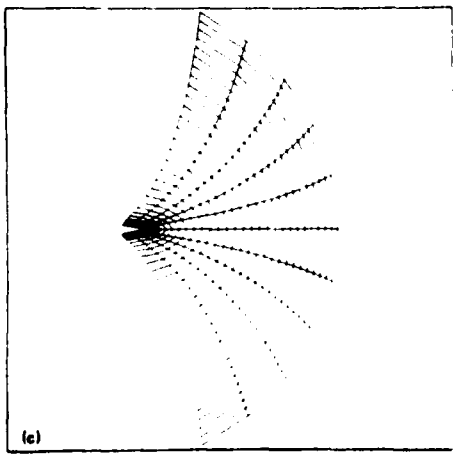


Fig. 5 Grid about wing having airfoil section and elliptical planform
(a) Closeup of innermost five circumferential lines in equatorial plane
(b) Closeup of trailing edge in equatorial plane
(c) Far-field behind trailing edge

CONCLUSION

An elliptic partial differential grid-generation technique which gives the user ability to control mesh cell size and skewness at a boundary has been generalized from two to three dimensions. The method discussed in this paper has been applied successfully to a variety of topologies and test cases. Future plans include substituting a faster solution method such as ADI, investigating the application of the method to a wider collection of topologies, and writing a transportable, user-oriented three-dimensional code, such as GRAPE is in two dimensions.

REFERENCES

1. Kutler, P., "A Perspective of Theoretical and Applied Computational Fluid Dynamics," AIAA Paper 83-0037, Presented at 21st Aerospace Sciences Meeting, Reno, Nevada, Jan. 10-13, 1983.

2. Winslow, A. M., "Numerical Solution of the Quasilinear Poisson Equation in a Nonuniform Triangle Mesh," J. Comp. Phys., Vol. 1, No. 2, Nov. 1967, pp. 149-172.

3. Chu, W. H., "Development of a General Finite Difference Approximation for a General Domain. Part I: Machine Transformation," J. Comp. Phys., Vol. 8, No. 3, Dec. 1971, pp. 392-408.

4. Godunov, S. K. and Prokopov, G. P., "The Use of Moving Meshes in Gas-Dynamical Computations," USSR Comp. Math. Phys., Vol. 12, No. 2, 1972, pp. 182-195.

5. Thompson, J. F., Thames, F. C., Mastin, C. M., "Automatic Numerical Generation of Body-Fitted Curvilinear Coordinate System for Field Containing Any Number of Arbitrary Two-Dimensional Bodies," J. Comp. Phys., Vol. 15, No. 3, July 1974, pp. 299-319.

6. Thompson, J. F., "General Curvilinear Coordinate Systems," Numerical Grid Generation, Thompson, J. F. ed., Elsevier Science, New York, 1982, pp. 1-30.

7. Steger, J. L. and Sorenson, R. L., "Automatic Mesh-Point Clustering Near a Boundary in Grid Generation with Elliptic Partial Differential Equations," J. Comp. Phys., Vol. 33, No. 3, Dec. 1979, pp. 405-410.

8. Sorenson, R. L., "A Computer Program to Generate Two-Dimensional Grids about Airfoils and Other Shapes by the Use of Poisson's Equation," TM-81198, May 1980, NASA/Ames Research Center, Moffett Field, CA.

9. Thompson, J. F. ed., Numerical Grid Generation, Elsevier Science, New York, 1982.

10. Mastin, C. W. and Thompson, J. F., "Transformations of Three-Dimensional Regions Onto Rectangular Regions by Elliptic Systems," Numerische Mathematic, Vol. 29, Fasc. 4, 1978, pp. 397-407.

11. Thames, F. C., "Generation of Three-Dimensional Boundary-Fitted Coordinate Systems for Wing/Wing-Tip Geometries Using the Elliptic Solver Method," Numerical Grid Generation, Thompson, J. F. ed., Elsevier Science, New York, 1982, pp. 695-716.

12. Shieh, C. F., "Three-Dimensional Grid Generation using Poisson Equations," Numerical Grid Generation, Thompson, J. F. ed., Elsevier Science, New York, 1982, pp. 687-694.

13. Thomas, P. D., "Numerical Generation of Composite Three Dimensional Grids by Quasilinear Elliptic Systems," Numerical Grid Generation, Thompson, J. F. ed., Elsevier Science, New York, 1982, pp. 667-686.

14. Pulliam, T. H. and Steger, J. L., "Implicit Finite-Difference Simulations of Three-Dimensional Compressible Flow," AIAA J., Vol. 18, No. 2, Feb. 1980, pp. 159-167.

APPENDIX B

GENERATION OF THREE DIMENSIONAL BODY FITTED COORDINATES USING HYPERBOLIC PARTIAL DIFFERENTIAL EQUATIONS

INTRODUCTION

Body conforming curvilinear grids are often used in finite difference flow field simulations. One reason for this is that the application of boundary conditions can be simplified in finite difference calculations because grid lines coincide with boundary lines. This is especially important in high Reynolds number viscous flow simulation in which high flow gradients near the body surface must be resolved.

The task of generating a satisfactory body conforming coordinate system is not easy. The grids must not be too distorted, they should have smooth variation, and they should be clustered to flow field action regions – typically near boundary surfaces. Moreover, the grids should be generated in an automatic manner that requires a minimum of user input.

One approach for generating body conforming grids with minimum user input has been to solve a set of partial differential equations. In this technique level lines of $\xi(x,y,z)$, $\eta(x,y,z)$, and $\varsigma(x,y,z)$ that have monotone variation are sought as a solution of a set of partial differential equations. Generally values of ξ, η and ς are user specified on the boundary surface and constraints expressed as differential equations are used to develop the grid away from the boundaries. The most popular such approach requires the solution of a set of elliptic equations that satisfy the maximum principle, however, hyperbolic and parabolic governing equations have been used as well, at least in two dimensional applications.

In this appendix one way of extending the hyperbolic grid generation method of Steger and Chaussee to three dimensions is developed. In two dimensions the two differential constraints

$$\xi_x \eta_x + \xi_y \eta_y = 0 \tag{1a}$$

$$\xi_x \eta_y - \xi_y \eta_x = (\Delta V)^{-1} \tag{1b}$$

or in ξ, η computational space

$$x_\xi x_\eta + y_\xi y_\eta = 0 \tag{2a}$$

$$x_\xi y_\eta - x_\eta y_\xi = \Delta V \tag{2b}$$

have been solved by marching in η from an initial data plane $\eta(x, y) =$ constant. The first equation is a constraint of orthogonality. The second equation controls the mesh spacing with the user specifying the mesh control volume ΔV (actually area in two dimensions). A linearized version of equations (2) is readily shown to be hyperbolic and suitable for marching in η. Equations (2) are solved in computational space to give the x, y location of the $\xi =$ constant and $\eta =$ constant grid lines.

The two partial differential equation , ..pressed as either Equations (1) or Equations (2), have been referred to as a mesh cell volume procedure for grid generation. In the next section a three dimensional procedure is developed.

THREE DIMENSIONAL GRID GENERATION EQUATIONS

A body fitted exterior grid about an arbitrary closed boundary surface is desired. Only a simple topology such as that illustrated in Figure (1) will be considered here. The body surface is chosen to coincide with $\varsigma(x,y,z) = 0$ and the surface grid line distributions of ξ = constant and η = constant are user specified. The outer boundary $\varsigma(x,y,z) = \varsigma_{max}$ is not specified, it is only required to be sufficiently far removed from the inner boundary. Using ς as the marching direction, partial differential equations are sought which produce planes of constant ξ, η and ς to form a nonsingular mesh system.

An extension of the mesh cell volume procedure to three dimensions is proposed. In three dimensions, however, there are three orthogonality relations and one cell volume constraint. At any point four equations are available to predict the three unknowns x, y and z so one equation must be discarded. Because ς is the marching direction it is natural to use only the two orthogonality relations that involve ς, this leads to the governing equations

$$x_\xi x_\varsigma + y_\xi y_\varsigma + z_\xi z_\varsigma = 0 \tag{3a}$$

$$x_\eta x_\varsigma + y_\eta y_\varsigma + z_\eta z_\varsigma = 0 \tag{3b}$$

$$x_\xi y_\eta z_\varsigma + x_\varsigma y_\xi z_\eta + x_\eta y_\varsigma z_\xi - x_\xi y_\varsigma z_\eta - x_\eta y_\xi z_\varsigma - x_\varsigma y_\eta z_\xi = \Delta V \tag{3c}$$

or with $\vec{r}$ defined as $(x,y,z)^t$

$$\vec{r}_\xi \cdot \vec{r}_\varsigma = 0, \quad \vec{r}_\eta \cdot \vec{r}_\varsigma = 0, \quad \left|\frac{\partial(x,y,z)}{\partial(\xi,\eta,\varsigma)}\right| = J^{-1} = \Delta V$$

The first two equations represent orthogonality relations between ξ and ς and between η and ς, while the last equation is the volume or finite Jacobian constraint.

Equations (3) comprise a system of nonlinear partial differential equations in which x, y and z are specified as initial data at $\varsigma = 0$. As developed below, linearization and

analysis of Equations (3) about a nearby known state reveals that the system is hyperbolic with ς as the marching direction.

Let x^0, y^0, z^0 represent a nearby known state so that

$$\begin{aligned} x &= x^0 + (x - x^0) = x^0 + \tilde{x} \\ y &= y^0 + \tilde{y} \\ z &= z^0 + \tilde{z} \end{aligned} \tag{4}$$

where $\tilde{x}, \tilde{y}$ and $\tilde{z}$ are small. Substitution of these expressions into Equations (3) and elimination of products of tilde terms results in the locally linearized system

$$A_0(\vec{r} - \vec{r}_0)_\xi + B_0(\vec{r} - \vec{r}_0)_\eta + C_0(\vec{r} - \vec{r}_0)_\varsigma = \vec{f} \tag{5}$$

with

$$A = \begin{pmatrix} x_\varsigma & y_\varsigma & z_\varsigma \\ 0 & 0 & 0 \\ (y_\eta z_\varsigma - y_\varsigma z_\eta) & (x_\varsigma z_\eta - x_\eta z_\varsigma) & (x_\eta y_\varsigma - x_\varsigma y_\eta) \end{pmatrix} \tag{6a}$$

$$B = \begin{pmatrix} 0 & 0 & 0 \\ x_\varsigma & y_\varsigma & z_\varsigma \\ (y_\varsigma z_\xi - y_\xi z_\varsigma) & (x_\xi z_\varsigma - x_\varsigma z_\xi) & (x_\varsigma y_\xi - x_\xi y_\varsigma) \end{pmatrix} \tag{6b}$$

$$C = \begin{pmatrix} x_\xi & y_\xi & z_\xi \\ x_\eta & y_\eta & z_\eta \\ (y_\xi z_\eta - y_\eta z_\xi) & (x_\eta z_\xi - x_\xi z_\eta) & (x_\xi y_\eta - x_\eta y_\xi) \end{pmatrix} \tag{6c}$$

$$\vec{r} = \begin{pmatrix} x \\ y \\ z \end{pmatrix}, \quad \vec{f} = \begin{pmatrix} -\frac{\partial \vec{r}}{\partial \xi} \cdot \frac{\partial \vec{r}}{\partial \varsigma} \\ -\frac{\partial \vec{r}}{\partial \eta} \cdot \frac{\partial \vec{r}}{\partial \varsigma} \\ \Delta V - \Delta V_0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \Delta V - \Delta V_0 \end{pmatrix} \tag{6d}$$

Let $\vec{R} \equiv \vec{r} - \vec{r}_0$, then (5) is rewritten as

$$A_0\vec{R}_\xi + B_0\vec{R}_\eta + C_0\vec{R}_\zeta = \vec{f} \tag{7}$$

Now C_0^{-1} exists unless $(\Delta V)^{-1} \to \infty$, which we will not impose, so (7) can be rewritten as

$$C_0^{-1}A_0\vec{R}_\xi + C_0^{-1}B_0\vec{R}_\eta + \vec{R}_\zeta = C_0^{-1}\vec{f} \tag{8}$$

Although the verification is nontrivial, $C_0^{-1}A_0$ and $C_0^{-1}B_0$ are found to be symmetric matrices (this was carried out by Dennis Jesperson of the NASA Ames Research Center, who used MACSYMA). The linearized system Equation (8) is therefore hyperbolic and can be marched with ζ serving as the "time-like" direction.

It can be pointed out that an analysis was attempted for the three orthogonality relations alone. These equations, however, are readily shown to be improperly posed for marching with initial data in ζ. Indeed, as best as we can discern, the three relations do not lend themselves to unique solutions regardless of the type of boundary conditions specified.

SOLUTION PROCEDURE

The nonlinear system of grid generation equations given by Equations (3) are solved with a noniterative implicit finite difference scheme. An unconditionally stable implicit scheme is chosen so the marching step size in ς can be arbitrarily selected based only on considerations of accurately generating the grid. Iterative solution of the nonlinear grid generation equations is avoided by expanding the equations about the previous marching step. As a consequence Equation (7) is solved with the nearby known state 0 taken from the previous ς step.

a) Numerical Method

Let $\Delta\xi = \Delta\eta = \Delta\varsigma = 1$ such that $\xi = j-1, \eta = k-1$ and $\varsigma = l-1$. Central spatial differencing of Equations (5) in ξ and η with first order backward implicit differencing in ς leads to

$$A_l\delta_\xi(\vec{r}_{l+1} - \vec{r}_l) + B_l\delta_\eta(\vec{r}_{l+1} - \vec{r}_l) + C_l \nabla_\varsigma \vec{r}_{l+1} = \vec{g}_{l+1} \tag{9}$$

where

$$\vec{g}_{l+1} = \begin{pmatrix} 0 \\ 0 \\ \Delta V_{l+1} \end{pmatrix}$$

and

$$\delta_\xi \vec{r}_j = \frac{\vec{r}_{j+1} - \vec{r}_{j-1}}{2}, \quad \delta_\eta \vec{r}_k = \frac{\vec{r}_{k+1} - \vec{r}_{k-1}}{2}$$

$$\nabla_\varsigma \vec{r}_{l+1} = \vec{r}_{l+1} - \vec{r}_l$$

Note that $C_l\delta_\varsigma\vec{r}_l$ was subtracted from $\vec{f}_{l+1}$ to produce $\vec{g}_{l+1}$ in theh above. Throughout only those indices that change are indicated, thus $r_{l+1} \Rightarrow r_{j,k,l+1}, r_{j+1} \Rightarrow r_{j+1,k,l}$ etc.

Multiplying through by C_l^{-1} gives

$$C_l^{-1}A_l\delta_\xi(\vec{r}_{l+1} - \vec{r}_l) + C_l^{-1}B_l\delta_\eta(\vec{r}_{l+1} - \vec{r}_l) + I(\vec{r}_{l+1} - \vec{r}_l) = C_l^{-1}\vec{g}_{l+1} \tag{10}$$

where I is the identifying matrix. To reduce the inversion cost the difference equations are approximately factored as

$$(I + C_l^{-1} A_l \delta_\xi)(I + C_l^{-1} B_l \delta_\eta)(\vec{r}_{l+1} - \vec{r}_l) = C_l^{-1} \vec{g}_{l+1} \tag{11}$$

so that $\vec{r}_{l+1}$ is obtained by solving sequences of one–dimensional–like block tridiagonal systems

$$(I + C_l^{-1} A_l \delta_\xi) \vec{g}^*_{l+1} = \vec{g}_{l+1} \tag{12a}$$

$$(I + C_l^{-1} B_l \delta_\eta) \bigtriangledown_\zeta \vec{r}_{l+1} = \vec{g}^*_{l+1} \tag{12b}$$

$$\vec{r}_{l+1} = \vec{r}_l + \bigtriangledown_\zeta \vec{r}_{l+1} \tag{12c}$$

Although not shown, numerical dissipation terms are added in ξ and η directions.

The coefficient matrices A_l, B_l and C_l contain ξ and η derivatives which are formed using central differences. These matrices also contain derivatives for x_ζ, y_ζ and z_ζ which are obtained from Equations (3) in terms of ξ and η derivatives. That is, Equations (3) are linear in the unknowns x_ζ, y_ζ and z_ζ. They are easily solved for as

$$\begin{pmatrix} x_\zeta \\ y_\zeta \\ z_\zeta \end{pmatrix} = \frac{\Delta V}{(DetC)} \begin{pmatrix} x_\xi z_\eta - y_\eta z_\xi \\ x_\eta z_\xi - x_\xi z_\eta \\ x_\xi y_\eta - x_\eta y_\xi \end{pmatrix} \tag{13}$$

with

$$Det(C) = (y_\xi z_\eta - y_\eta z_\xi)^2 + (x_\eta z_\xi - x_\xi z_\eta)^2 + (x_\xi y_\eta - x_\eta y_\xi)^2$$

Note that $\Delta V / \sqrt{(x_\zeta^2 + y_\zeta^2 + z_\zeta^2)} = \mathrm{Det}(C)$ so that $\mathrm{Det}(C)$ will be zero if and only if the user specified $\Delta V = 0$. Hence, C^{-1} will exist.

b) Cell Volume Specification

The user has control of the grid by means of the initial surface point distribution and by specification of the cell volumes, $\Delta V_{j,k,l}$. Through the cell volumes the extent and clustering of the grid can be essentially controlled. Because the cell volume at each point must be specified, it is clear that the user must devise some kind of method for determining volumes. There are many possibilities, here one such approach is illustrated.

Suppose we had a sphere to grid. A reasonable grid might have uniform angle spacing and have a radial grid distribution that is exponential. For this special geometry and grid we can analytically determine the control volumes by a simple formula. Now take the problem at hand, perhaps an aircraft fuselage, which we want to mesh as a warped spherical-like grid. We can find a sphere that has the same surface area as our fuselage and use the grid cell volumes of the sphere to specify the cell volumes of the fuselage grid. However, the fuselage will not have the same kind of surface area distribution as a sphere with equal angle distribution. So here we need an adjustment, something like

$$\Delta V_{j,k,l} = (\Delta V_{j,k,l})_{sphere} \quad \left[(1-\delta) + \frac{(\Delta A_{j,k})_{sphere}}{(\Delta A_{j,k})_{fuselage}}\,\delta\right] \tag{14}$$

where $\delta \rightarrow 1$ for small l and $\delta \rightarrow 0$ for large l. That is, the volumes would be adjusted initially to the local boundary surface increments. But as we march out the uniform spherical volumes would gradually be specified. Such an approach has been used, and, as a result, the far field portion of the grid tends to be uniformly spherical. The results shown in the next section illustrate this behavior.

RESULTS

A series of plots are shown in Figures (1) – (4) to indicate the generation of grids about ellipsoids and cambered ellipsoid test geometries. Figure (1) sets the grid notation while Figures (2) – (4) show some typical results. The grids shown in Figure (2) are for an ellipsoid which has major to minor axis ratios of 4 to 1 and 2 to 1. The views 2a to 2c show the user specified surface point distributions from various observer positions. Similar views are shown in Figures (2d) to (2i) after marching 3 and 19 steps in the ζ direction. The views shown in Figure (3) show a grid generated about a cambered ellipsoid. Finally the views shown in Figure (4) show a grid about a wing–like ellipsoid ratioed 6 : 1 : 1/6.

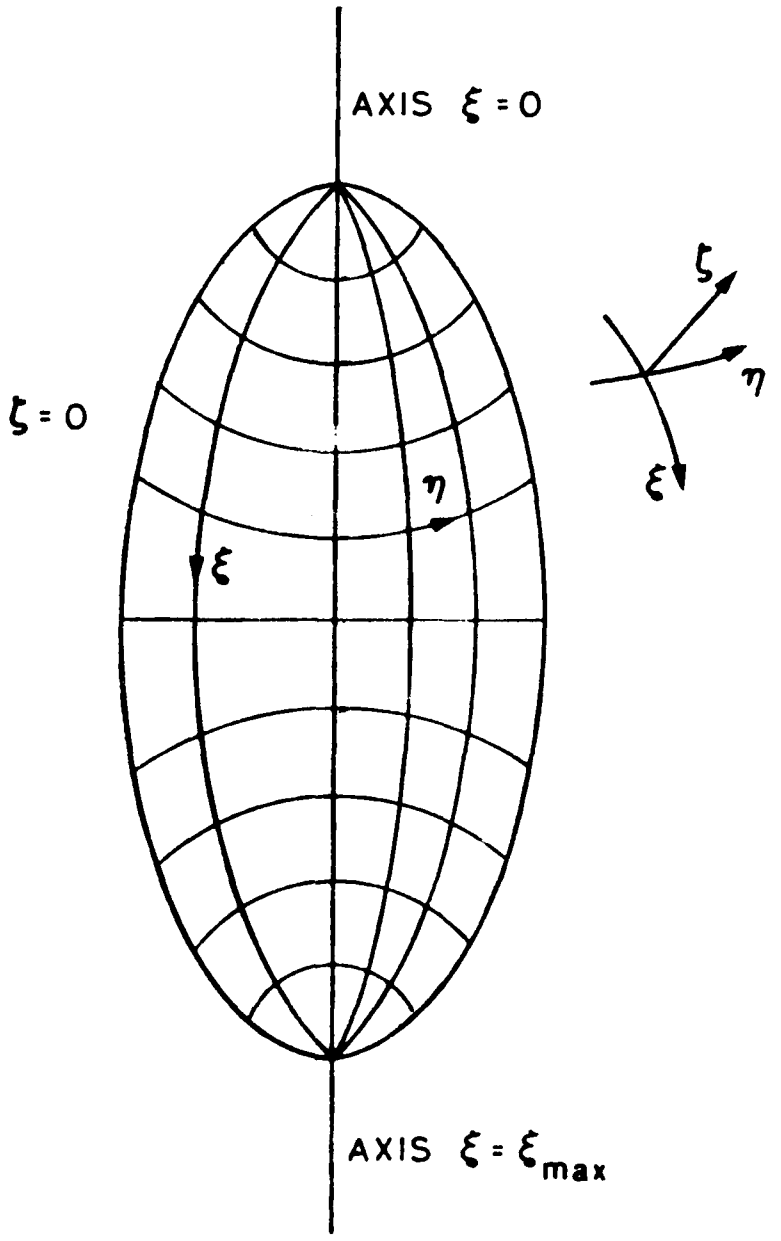


Figure 1. Surface Distribution at $\zeta = 0$

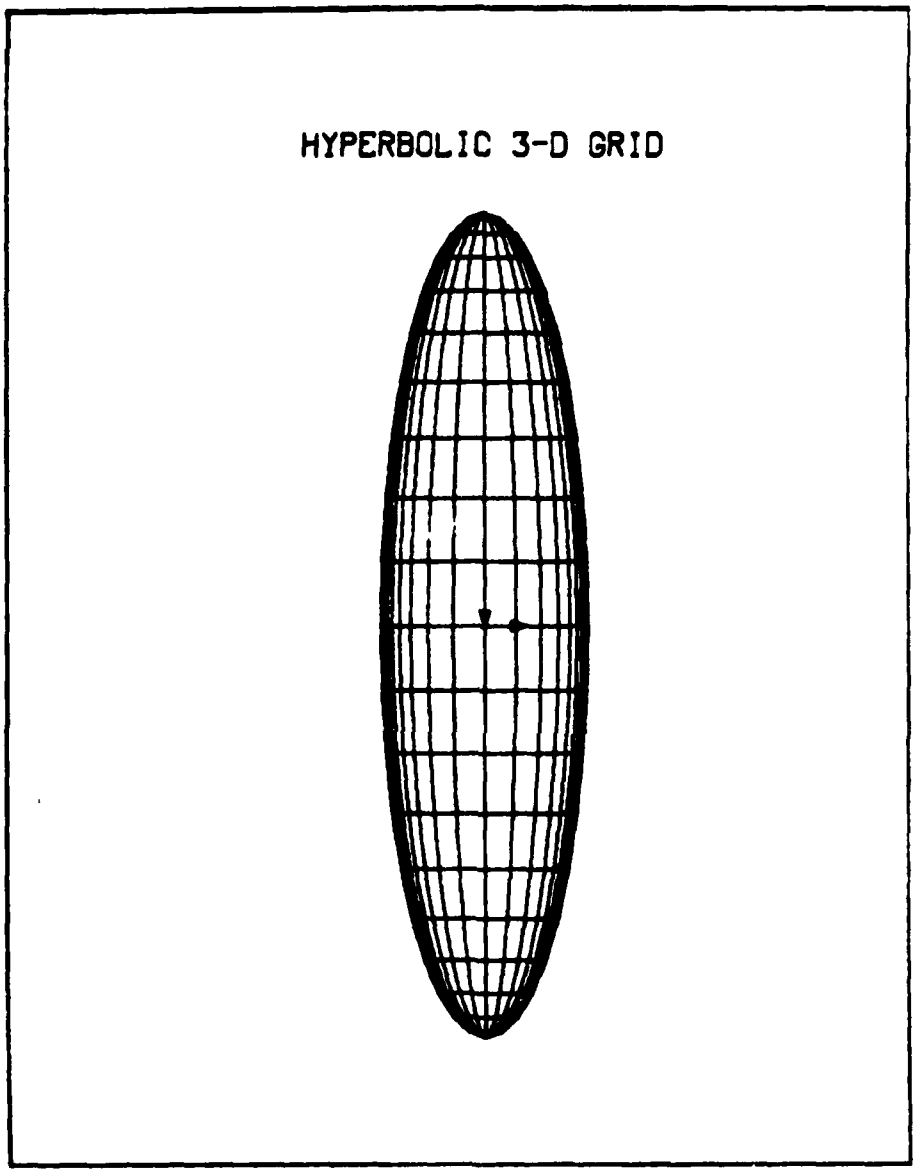


Figure 2a. Surface Distribution on Ellipsoid, $\zeta = 0$.
View at $\eta = 0$.

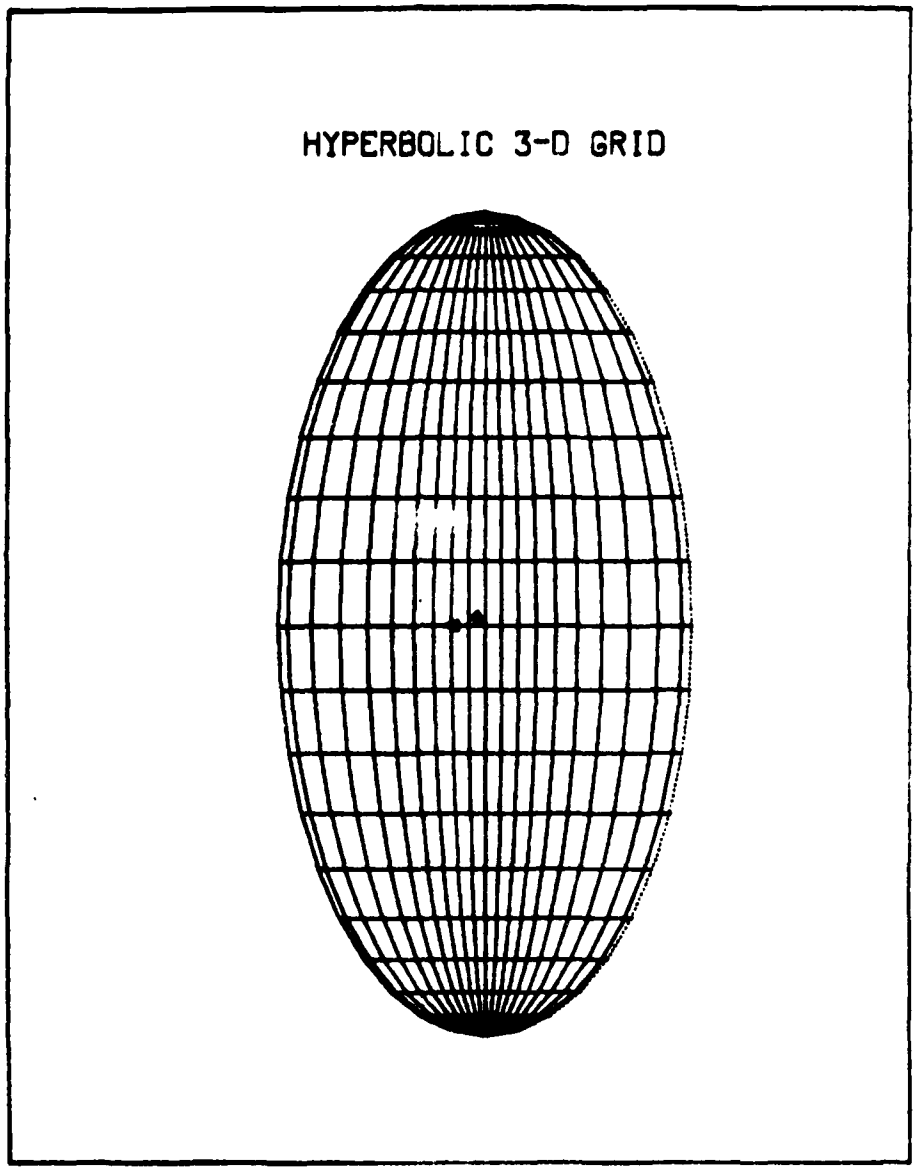


Figure 2b. Surface Distribution on Ellipsoid, $\varsigma = 0$.
View at $\eta \propto 90°$.

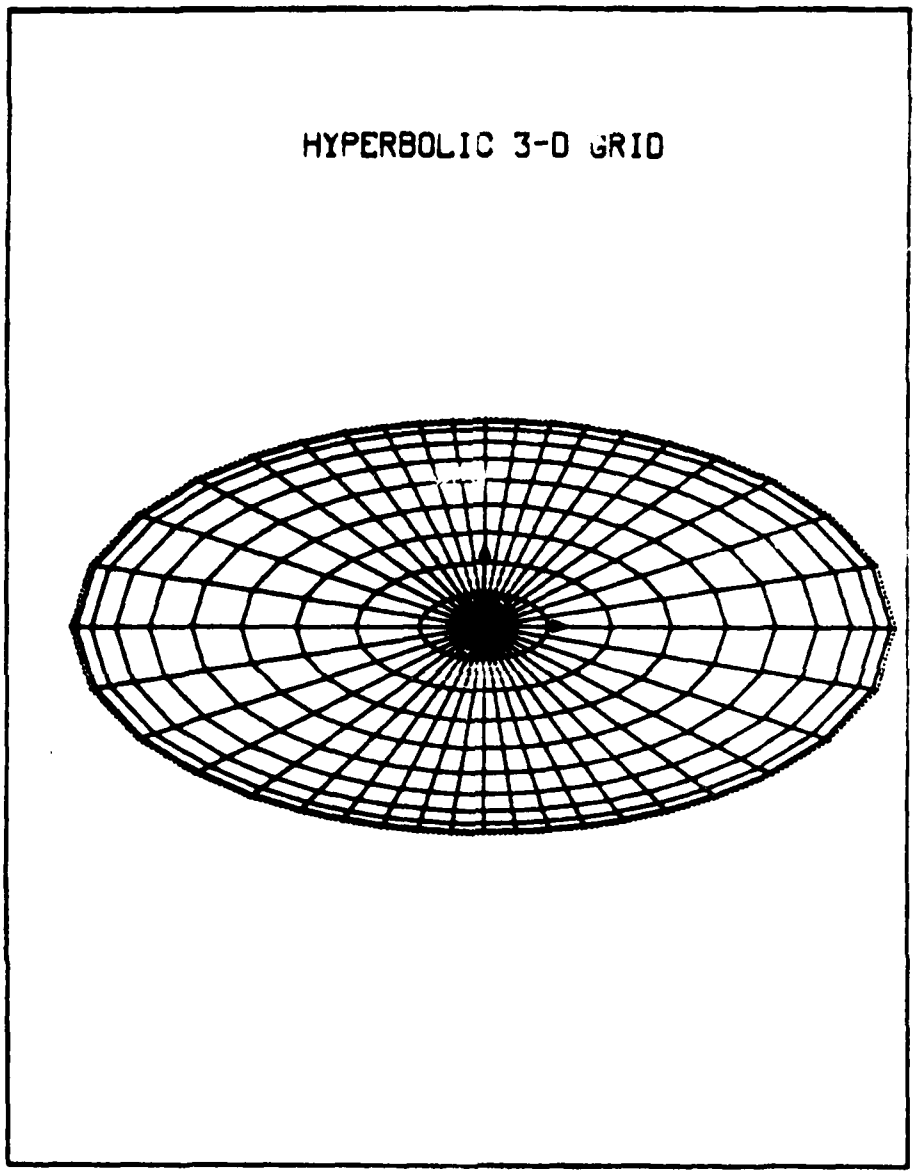


Figure 2c. Surface Distribution on Ellipsoid, $\zeta = 0$.
View from $\xi = 0$ "North Pole."

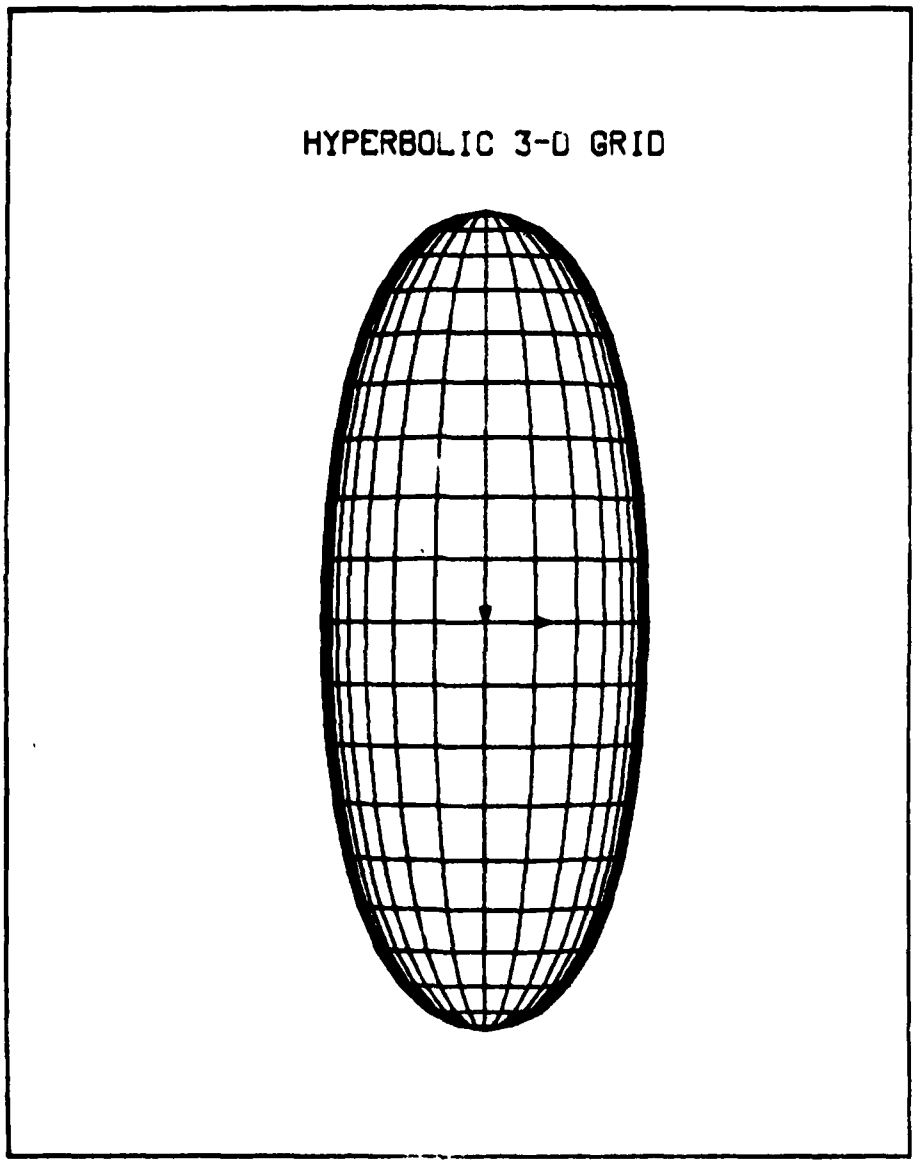


Figure 2d. Grid at $\zeta = 3\Delta\zeta$.
View at $\eta = 0$.

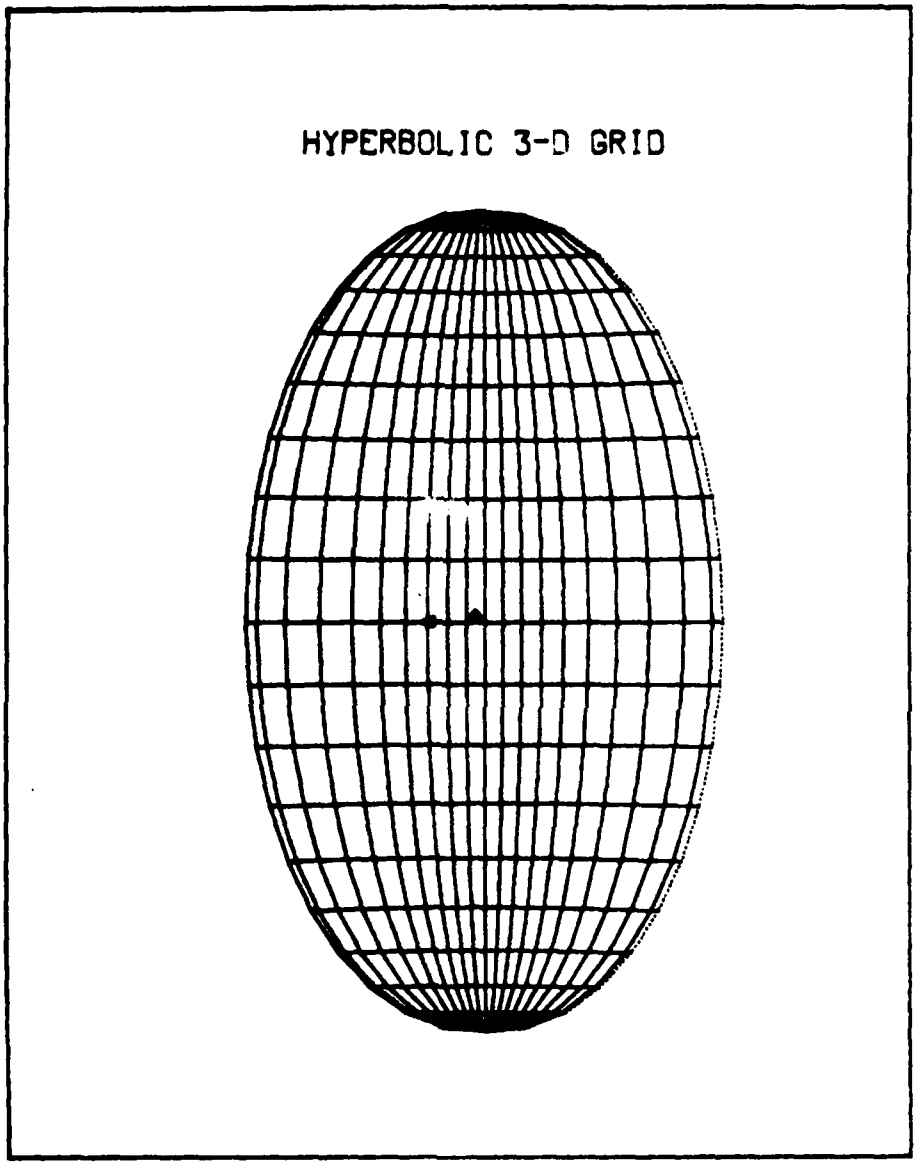


Figure 2e. Grid at $\zeta = 3\Delta\zeta$.
View at $\eta \propto 90°$.

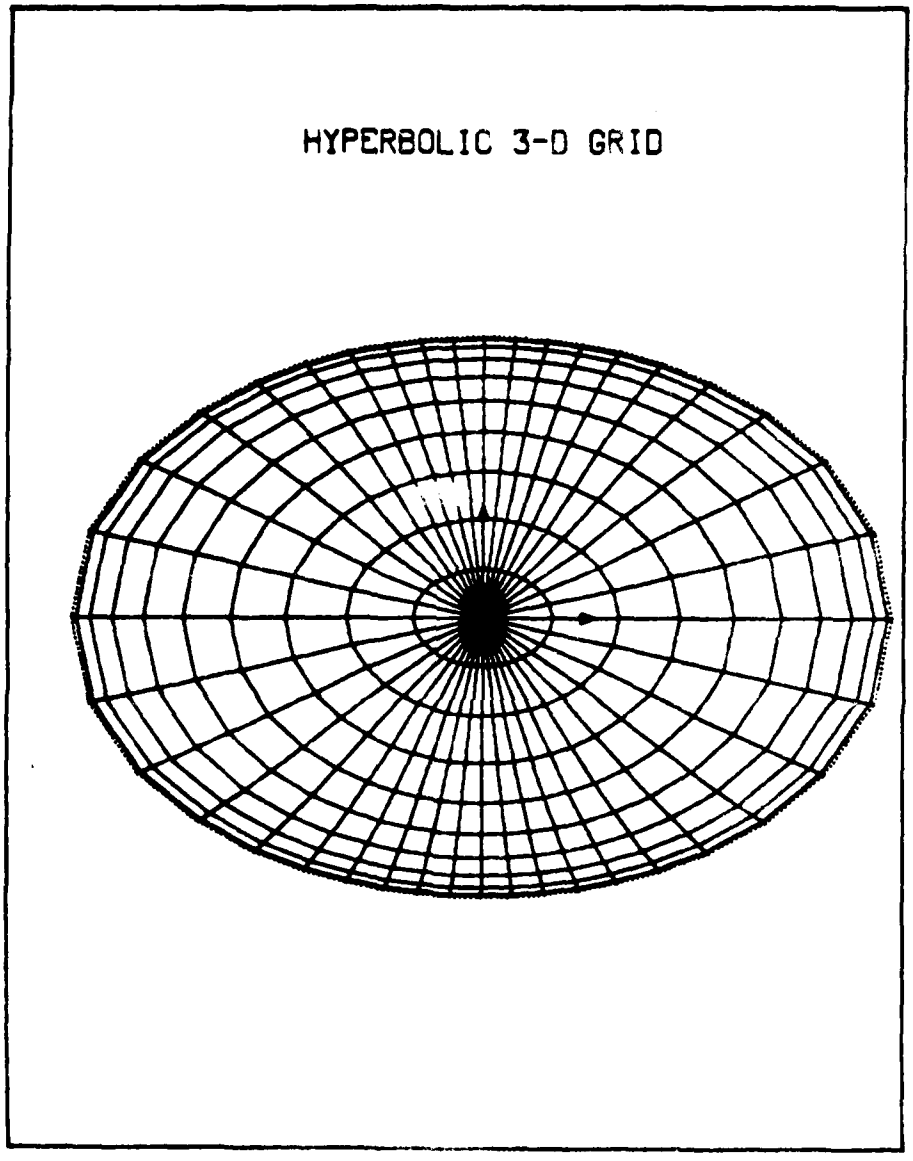


Figure 2f. Grid at $\varsigma = 3\Delta\varsigma$.
View at $\xi = 0$.

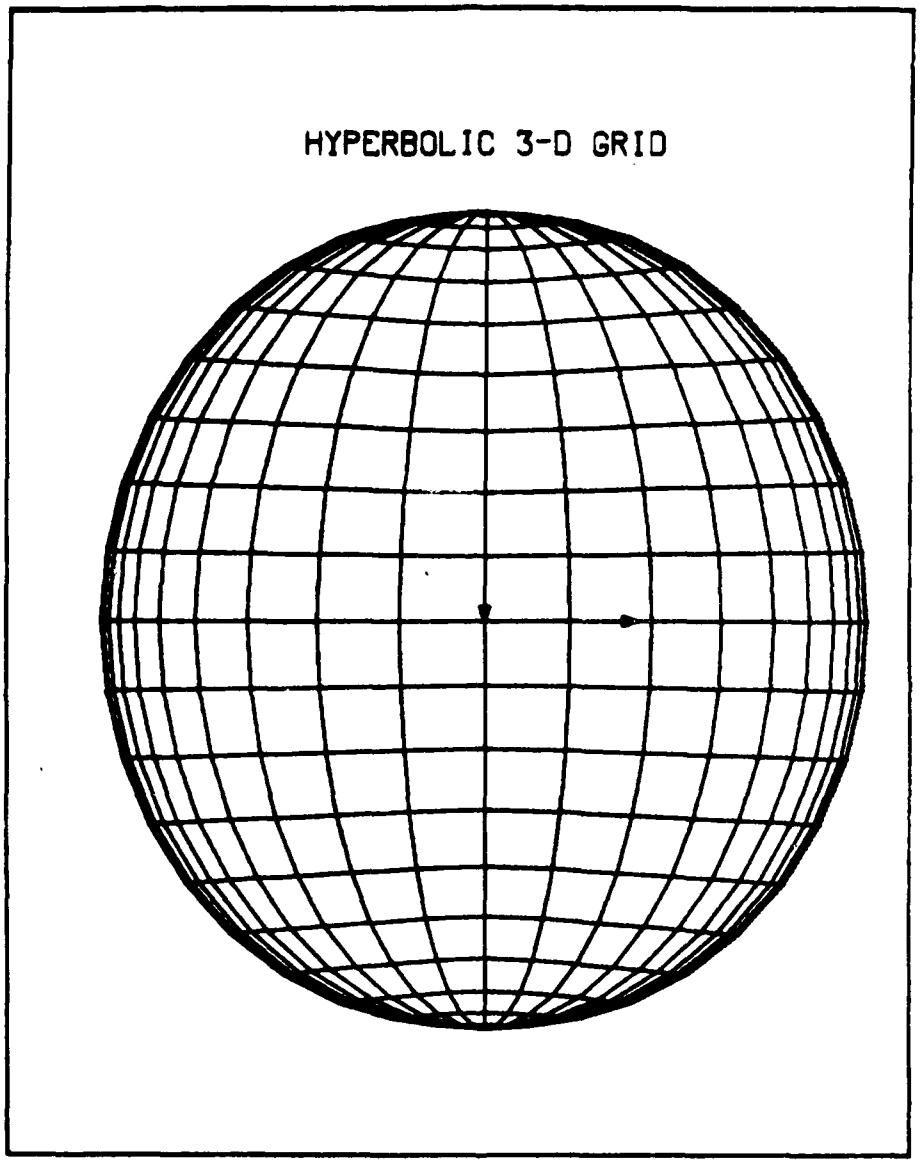


Figure 2g. Grid at $\zeta = 19\Delta\zeta$.
View at $\eta = 0$.

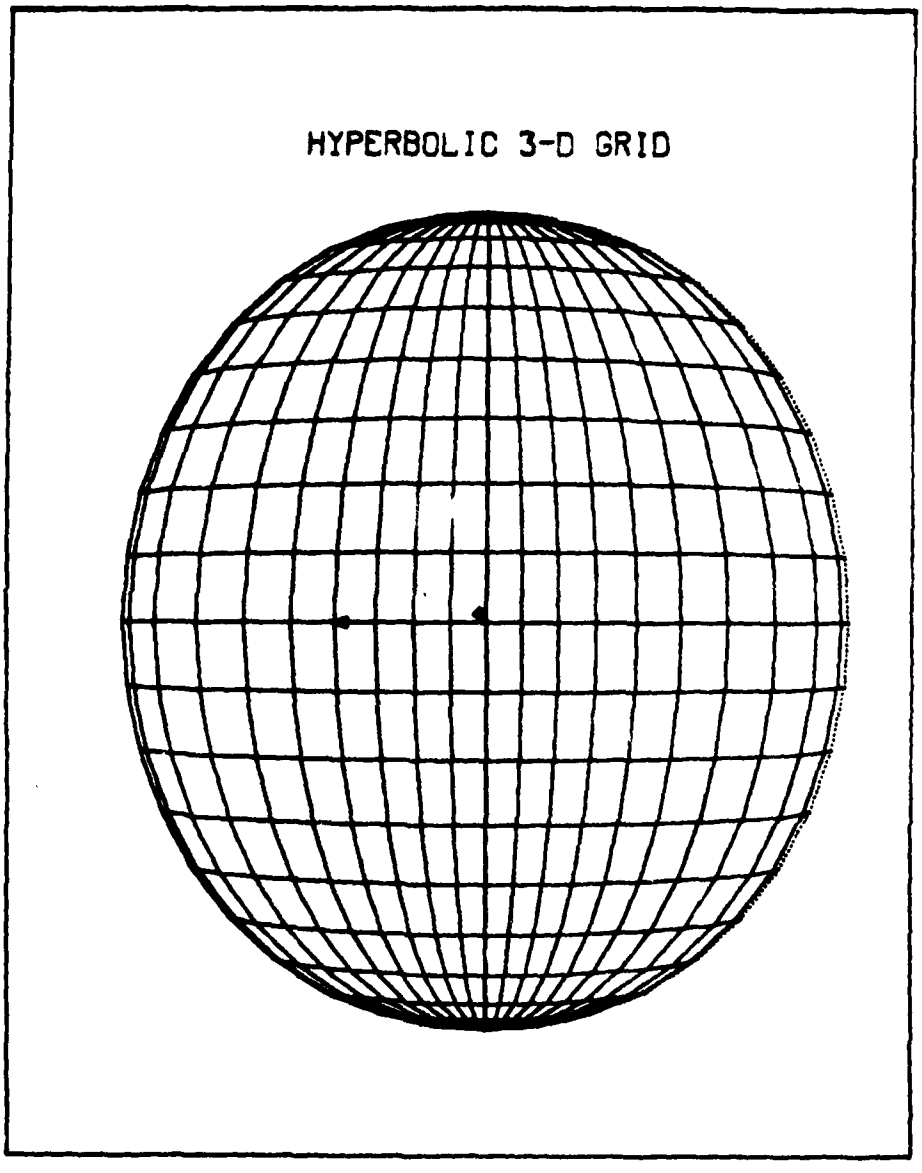


Figure 2h. Grid at $\varsigma = 19\Delta\varsigma$.
View at $\eta \propto 90°$.

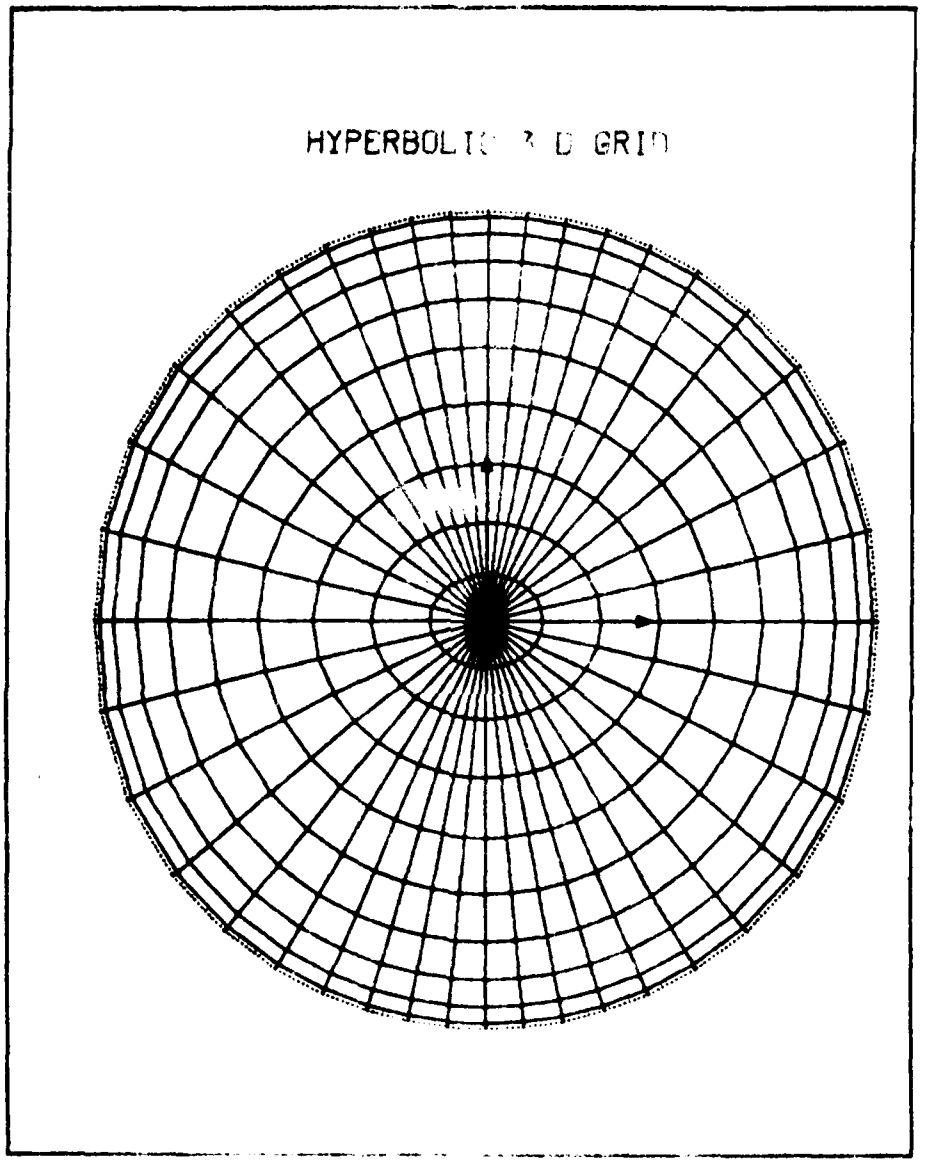


Figure 2: Grid at $\zeta = 19\Delta\zeta$.
View at $\zeta = 0$.

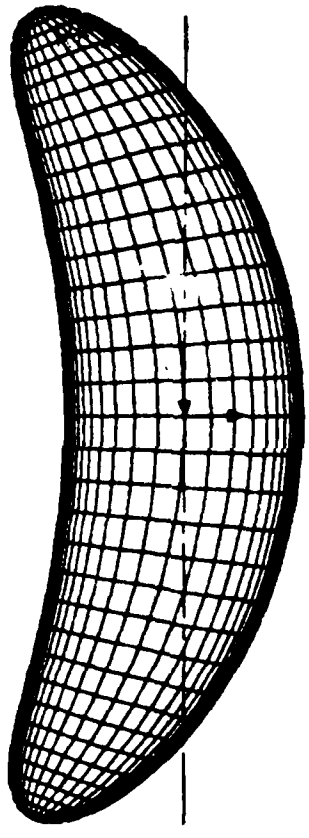


Figure 3a. Cambered Ellipsoid, $\varsigma = 0$.

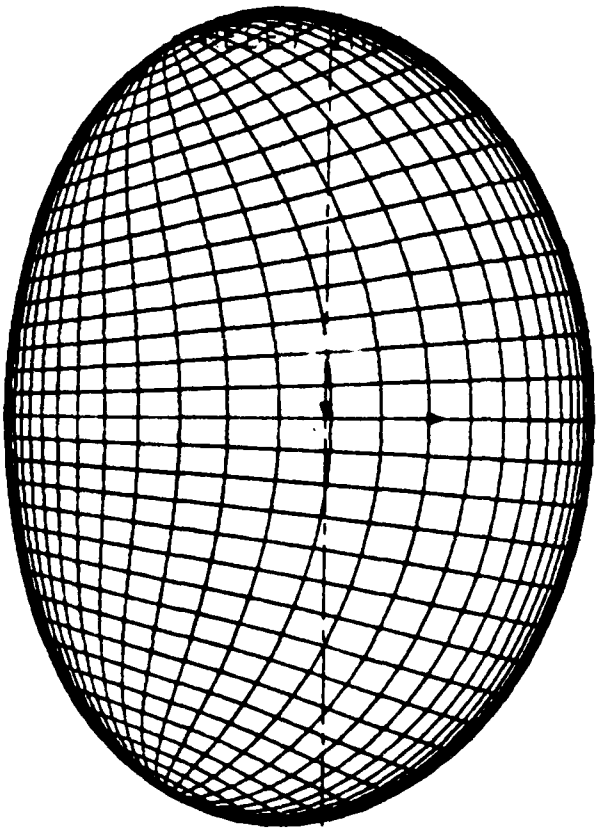


Figure 3b. Grid About Cambered Ellipsoid
At Surface $\varsigma = 19\Delta\varsigma$.

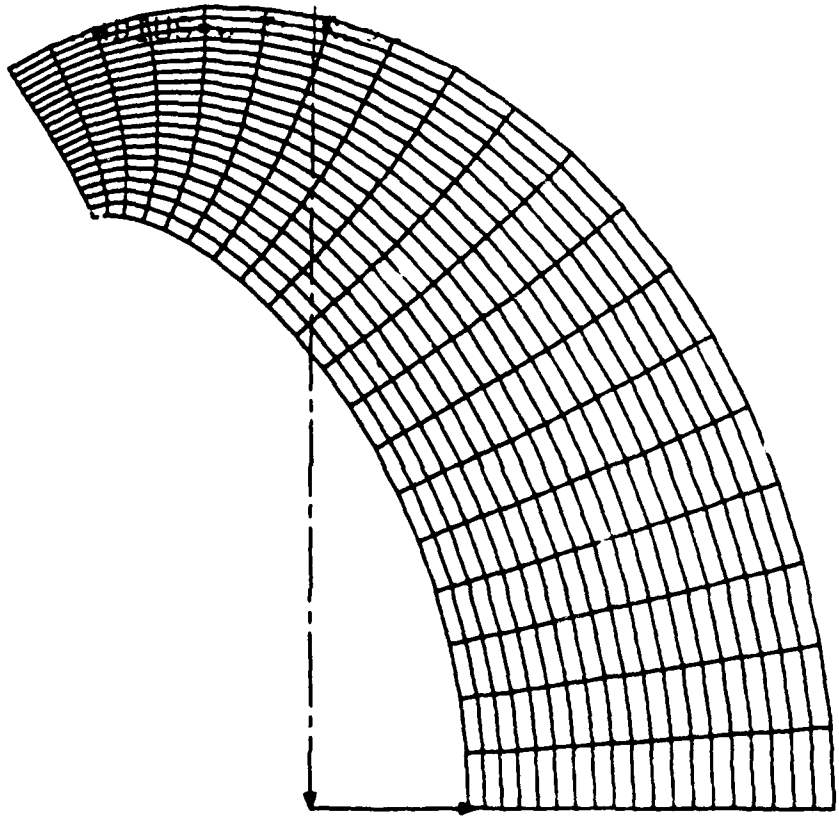


Figure 3c. Grid About Cambered Ellipsoid.
Side View, Upper Right Quarter.

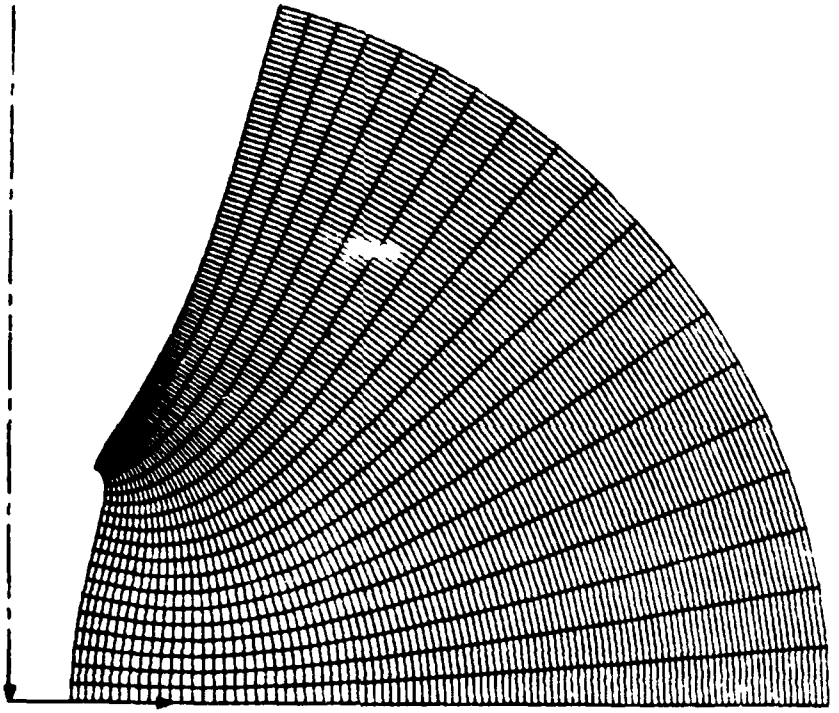


Figure 3d Grid About Cambered Ellipsoid.
Side View, Upper Left Quarter.

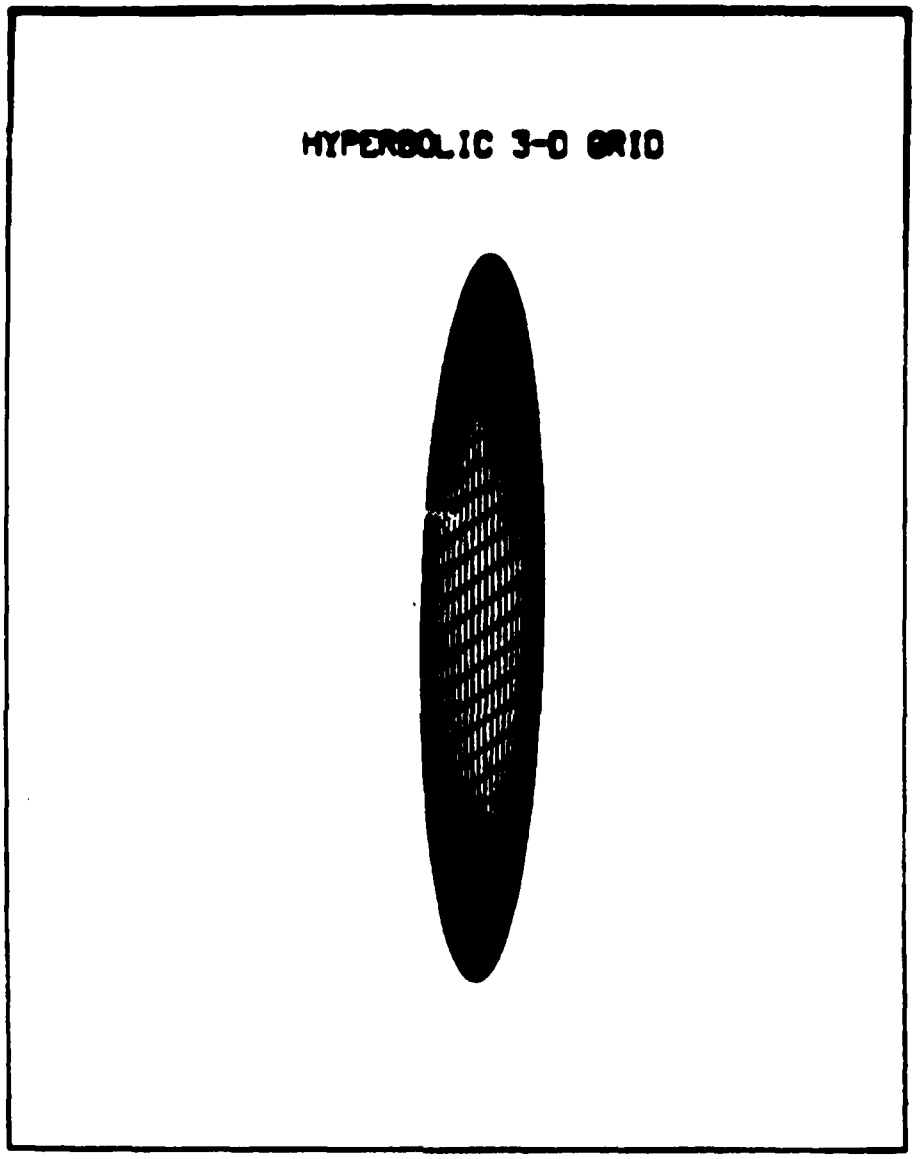


Figure 4a. Surface Distribution of 6 : 1 : 1/6 Ellipsoid
Span = 6, Chord = 1, Thickness = 1/6

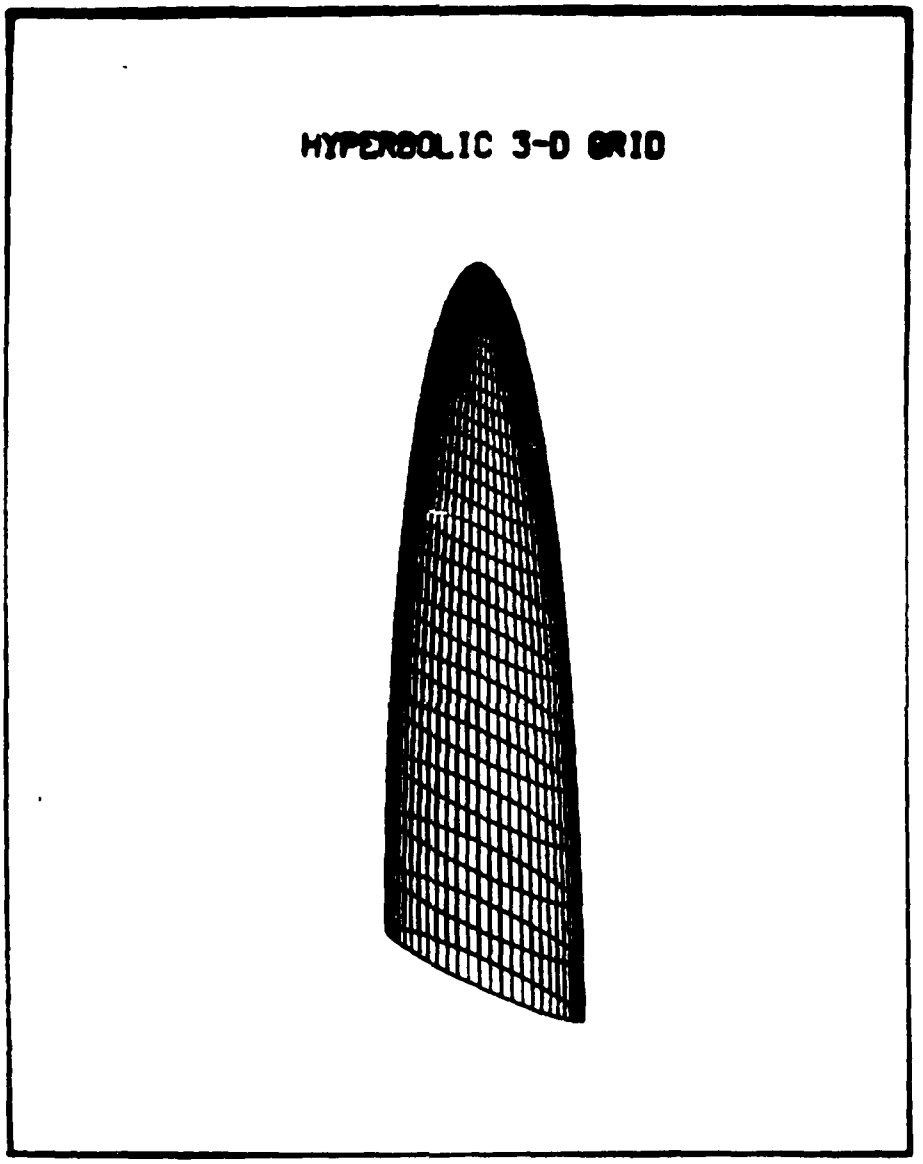


Figure 4b. Surface Distribution of 6 : 1 : 1/6 Ellipsoid
Span = 6, Chord = 1, Thickness = 1/6

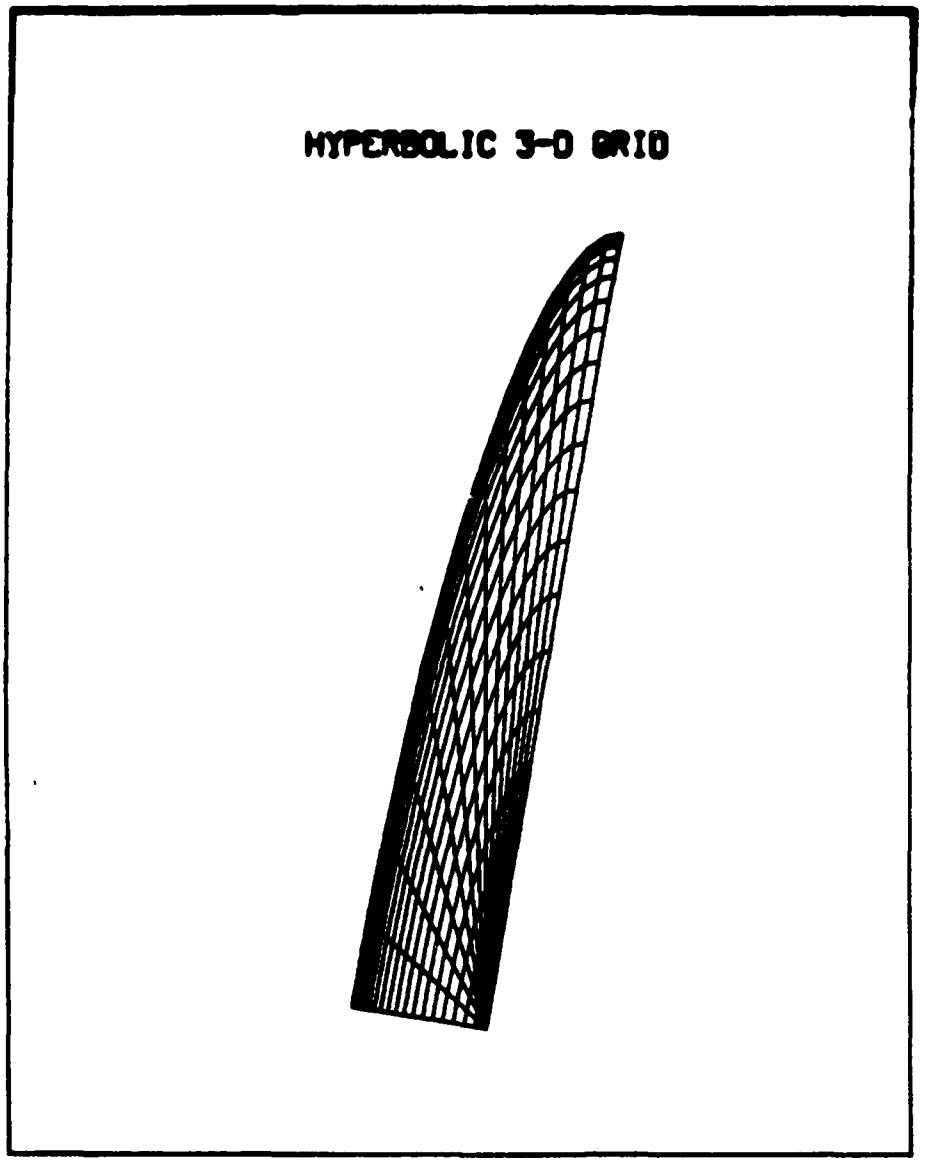


Figure 4c. North Pole View of 6 : 1 : 1/6 Surface Distribution

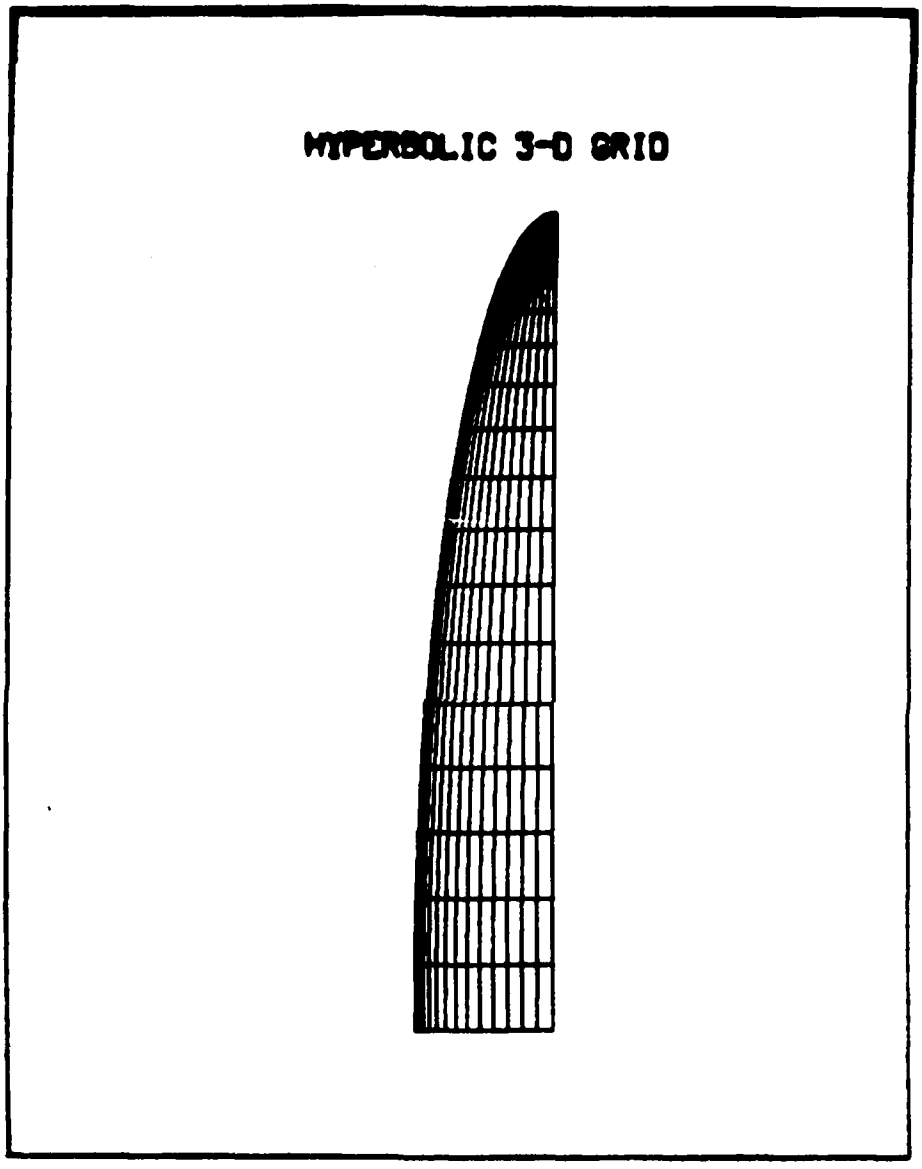


Figure 4d. Surface Distribution, Partial Planform View of 6 : 1 : 1/6

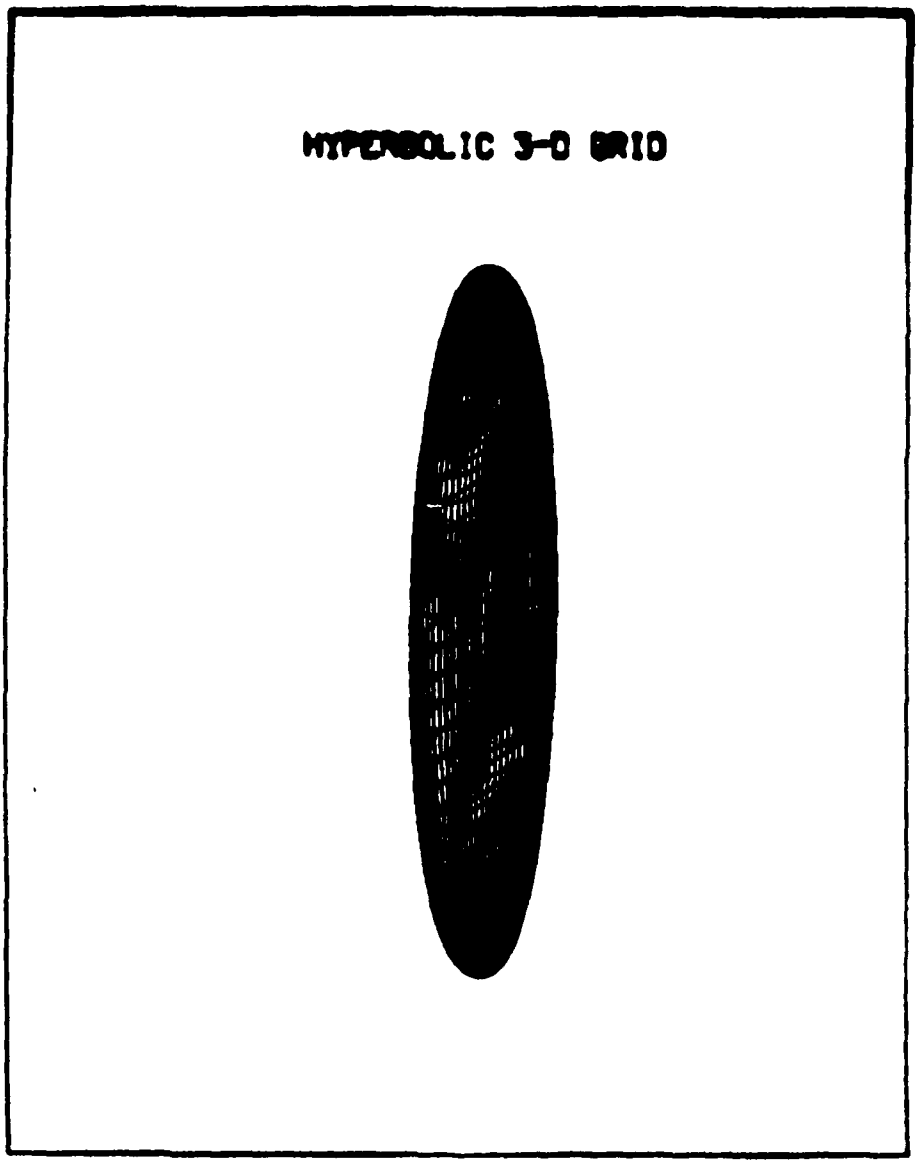


Figure 4e. Grid About 6 : 1 : 1/6 - 25th Grid Plane Out

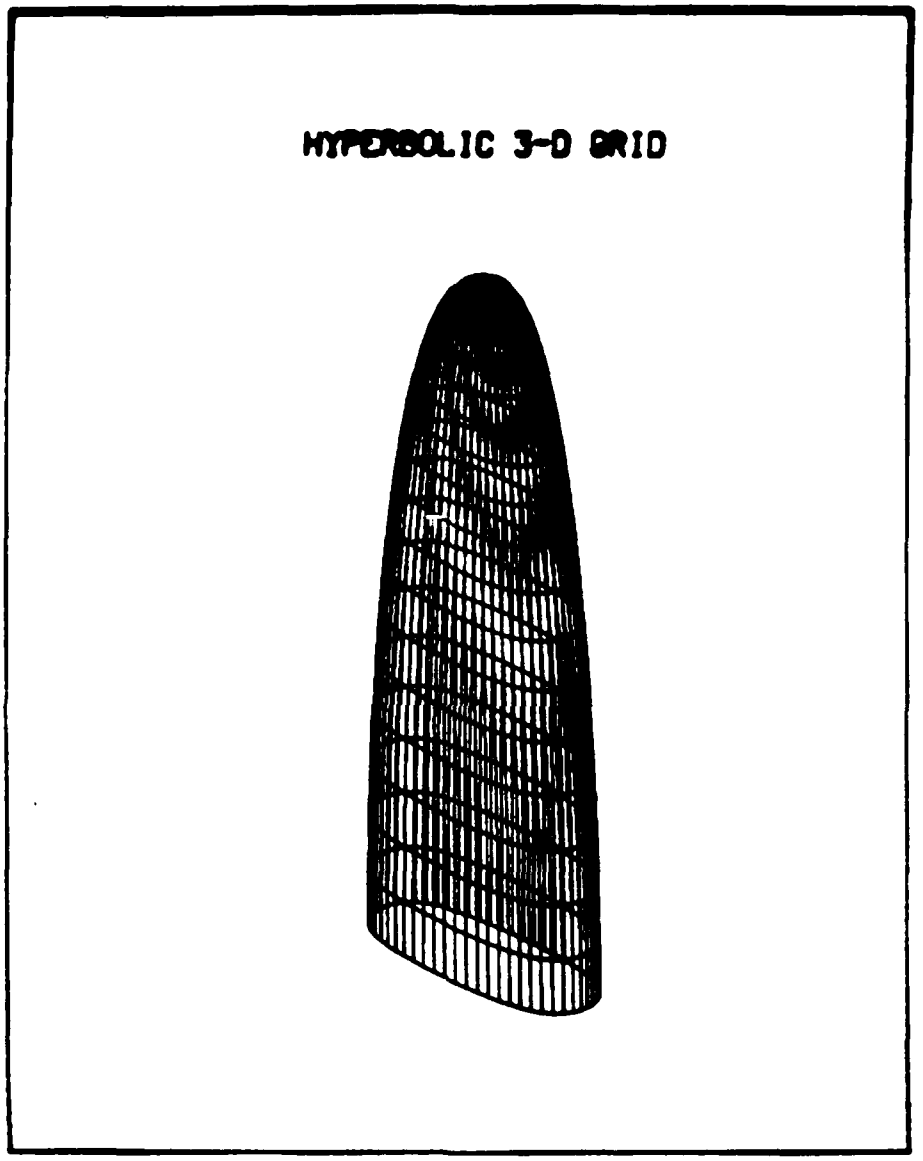


Figure 4f. 25th Grid Plane Out From 6 : 1 : 1/6

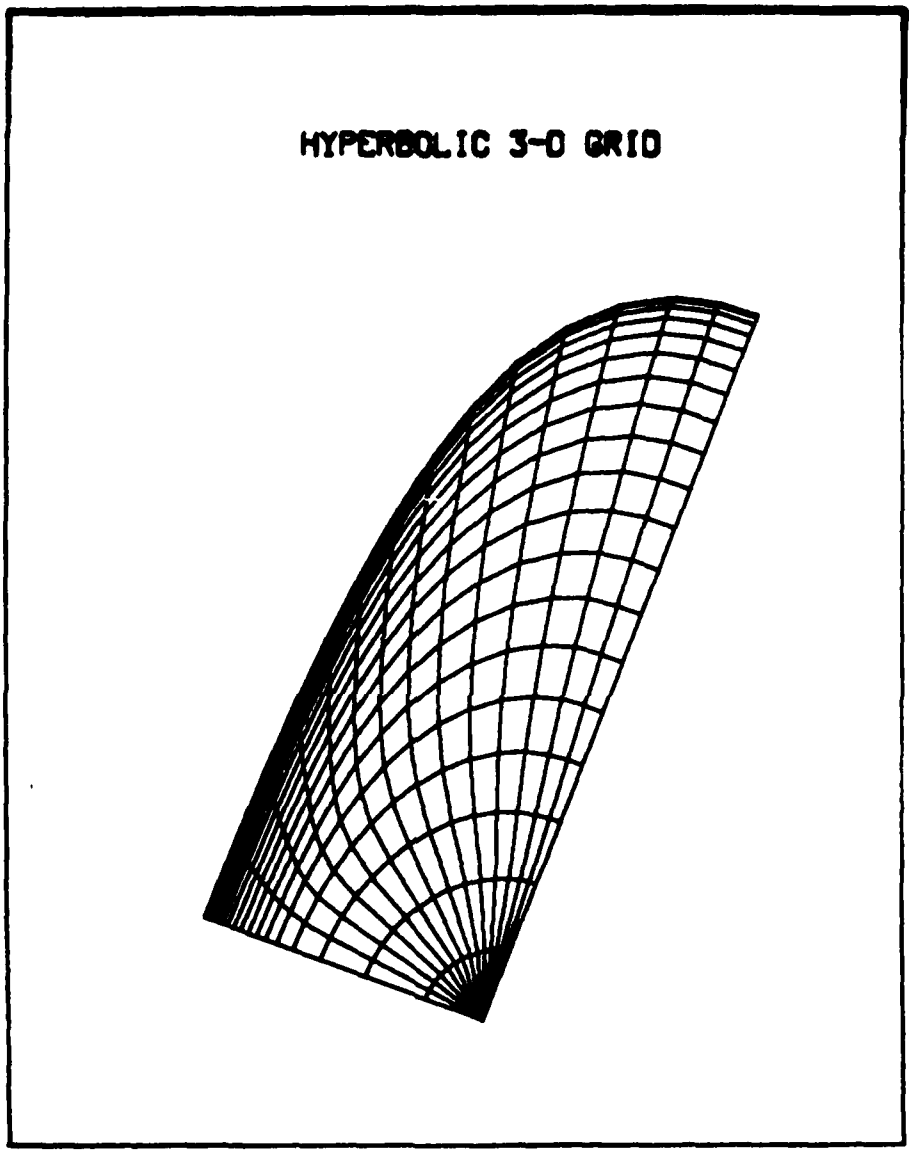


Figure 4g. North Pole View - 25th Grid Plane
Out From 6 : 1 : 1/6

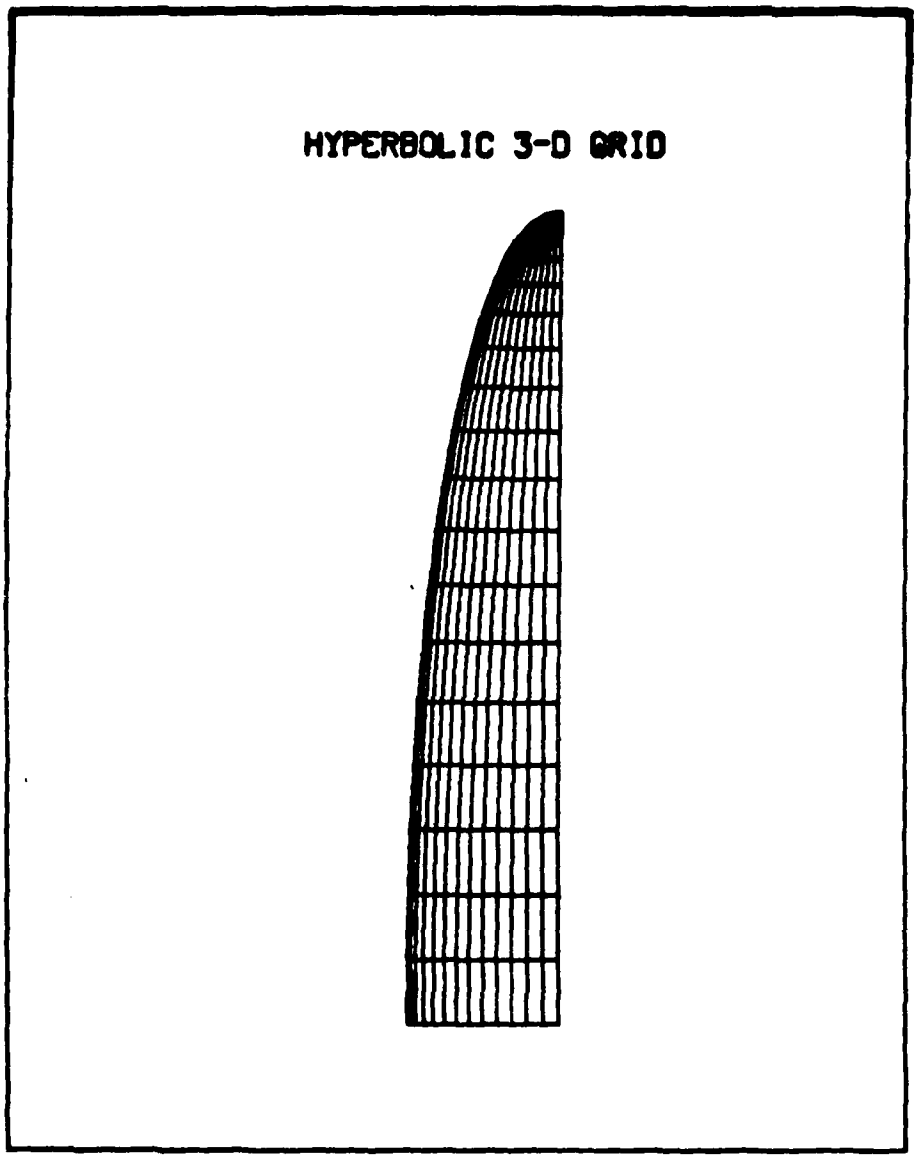


Figure 4h. 25th Grid Plane Out 6 : 1 : 1/6

APPENDIX C

A CHIMERA GRID SCHEME

Joseph L. Steger
Associate Professor
Department of Aeronautics and Astronautics
Stanford University
Stanford, California

F. Carrol Dougherty
Research Scientist
Applied Computational Aerodynamics Branch
NASA Ames Research Center
Moffett Field, California

John A. Benek
Research Scientist
Computational Fluid Dynamics Section
Calspan, AEDC
Tullahoma, Tennessee

ABSTRACT

A mesh system composed of multiple overset body-conforming grids is described for adapting finite-difference procedures to complex aircraft configurations. In this so-called "chimera mesh," a major grid is generated about a main component of the configuration and overset minor grids are used to resolve all other features. Methods for connecting overset multiple grids and modifications of flow-simulation algorithms are discussed. Computational tests in two dimensions indicate that the use of multiple overset grids can simplify the task of grid generation without an adverse effect on flow-field algorithms and computer code complexity.

INTRODUCTION

Finite-difference simulations of flow about realistic aircraft configurations are likely to use more than one grid system and more than one governing equation set. A composite computer code for such simulations is thus chimera-like — a chimera being a mythological creature with the head of a lion, the body of a goat, and the tail of a snake.

A finite-difference computer code that uses multiple equation sets (e.g., zones of Navier-Stokes and velocity potential) and multiple grids will be considerably more complicated than a code that uses a single flow-field equation set and a single grid mapping. Nevertheless, such a chimera code will lead to increased computational efficiency in flow simulation of complex aircraft configurations. As a result, the computational aerodynamicist will likely spend considerably more time in the future developing ways to interface grids, governing equations, and data bases.

The purpose of this paper is to explore a chimera-type mesh scheme in which a major grid is generated about a main body element such as a wing-fuselage, and minor grids are overset on the major grid so as to resolve secondary features of the configuration such as stores, nacelles, and canards. In general, the minor grids are overset on top of the major grid without requiring mesh boundaries to join in any special way. With the use of such an overset mesh system the task of grid generation is simplified because individual grids can be generated almost independently and then superimposed to form the overall mesh system. In this way, one can build up a grid system for treating complex configurations and avoid severe mesh distortions.

Overset mesh systems have been used previously. The explicit finite-difference code of Magnus and Yoshihara (1) for solving the transonic flow about airfoils is an early example of overset grids used to achieve computational efficiency. More recently, Berger and Oliger (2) have developed a numerical procedure which automatically inserts overset grids to resolve gradient regions of two-dimensional convection problems. As to applications to complex geometries, Starius (3) has used such an approach for the shallow-water equations, and, in work begun at Ames Research Center, Atta (4) has coded overset grids for the transonic potential equation. Atta has also extended the concept to three dimensions (5). We also have long argued the potential advantages for such a mesh system (6,7).

In the research described here we have undertaken an independent development. This is in order to adapt our grid system to an existing class of implicit finite-difference algorithms for solving the unsteady potential, Euler and thin-layer Navier-Stokes, and parabolized Navier-Stokes equations. These codes all use similar coordinate transformations, and zonal-flow-equation versions are under way. Ultimately we intend to explore the complexities of a true chimera code by combining the various equation sets with the overset grid scheme.

In this paper we develop some of the philosophy and details of the overset mesh system. The methodology here is restricted to two dimensions, and the present solutions, which utilize the stream function, are intended for grid evaluation. Overset grid calculations using the Euler and thin-layer Navier-Stokes equations will be presented elsewhere in a subsequent publication by Benek, Steger, and Dougherty.

APPROACH AND MOTIVATION

We wish to build a mesh system in which there is a major grid generated about a dominant body. Minor grid systems are generated about remaining portions of the body boundary or are used to better resolve portions of the major grid. At this point we do not allow the minor grids to communicate with each other except through the major grid. The sketches shown in Figs. 1 to 4 are presented with the discussion below to clarify these ideas.

Figure 1 illustrates an airfoil-flap combination. A major grid is generated about the main airfoil; a minor grid is used to resolve the flap. The two grids are not joined at a common boundary and so are connected as follows: data from the major grid are interpolated to supply outer boundary conditions to the minor grid. Thus, the flap grid receives input from the main airfoil. Within some curve of the minor grid that circumscribes the flap, points of the major grid will be blanked out. The major grid points forming a perimeter to the blanked region will also be flagged. Flow variables at these perimeter points are supplied by interpolating the minor grid solution. Thus the effect of the flap is imposed on the main airfoil.

The grids shown in Fig. 2 illustrate another application of overset grids. Here a main rectangular-like grid fits well to a cascade blade element everywhere except at the blunt nose region. A minor mesh, wrapped about the nose, is inserted to resolve the flow suction peak. The edge boundaries of the minor grid are interpolated from the major grid. Points in the major grid in the vicinity of the nose are blanked, and the flow values are found by interpolating data from the fine minor grid wrapped around the nose. This choice of overset grids can be preferable to the use of a C-grid throughout because a very skewed grid results for blades with high solidity, as indicated in Fig. 3.

Figure 4 illustrates multiple overset grids. In this case the minor grid resolving the airfoil leading-edge flap (slat) is shown intersecting the main airfoil. Such grid points must be turned off and a fringe boundary must be defined about these points with data supplied by interpolating the major grid.

A system of overset grids has many advantages. It can be used to treat complex geometries, resolve large flow-field gradients, and eliminate grid distortion. Overset grids are easier to construct than grids generated by patching meshes together at common boundaries because each grid is somewhat independently generated. This independence ensures that each grid will maintain a well-ordered set of points that are successively indexed. Well-orderedness is an important property of a mesh because when finite-difference approximations are applied to it, a set of structured algebraic equations results. The computational efficiencies gained by using, for example, spectral methods, approximate factorization or alternating directions techniques, and vectorized computer programming require structured matrices. Another advantage of overset grids is a natural occurrence of overlapping grid points between interpolated boundaries. If the grids are sufficiently well overlapped, the stability of an implicit algorithm should not be adversely affected, even though the interpolated boundary-values are updated in an explicit mode. For a similar reason, iterative convergence can be improved by overlapping mesh boundaries (8).

There are also disadvantages to the overset mesh system. Interpolation points and blanked points must be located and labeled for special treatment. The solution algorithm also becomes more complex relative to using a single grid mapping. Finally, interpolation can cause inaccuracies such as local loss of conservative form.

ALGORITHM CHANGES

To use overset grids the finite-difference algorithms need only be altered in three essential ways. First, the data base must be structured for a number of grids, each of which can have a different dimension. This is easily accomplished in a variety of ways depending on the computer. For example, on a single instruction-stream machine, use of a single array can be practical. In this case the index j,k of the mth grid can be located as a single array-point i. The FORTRAN variable Q(I) is located by the index

$$I = J + (K - 1) * JMAX(M) + ISUM(M)$$

where JMAX(M) and KMAX(M) are maximum j and k values of the Mth grid and ISUM(M) is the number of grid points in all grids before the Mth grid. Using such a single index leads to a dependency which does not vectorize on some machines, in which case other constructions should be used.

Second, the algorithm must skip blanked points. In time-like marching algorithms this is easy to implement by simply multiplying by a 0 or 1 flag array stored at each point. Such blanking is illustrated below for the centrally differenced Laplacian in which an implicit approximately factored delta-form algorithm is used as part of an iterative solution process. That is,

$$\delta_{xx}\psi^n_{j,k} + \delta_{yy}\psi^n_{j,k} = 0 \tag{1}$$

where

$$\left.\begin{aligned}\delta_{xx}\psi_j &= (\psi_{j+1} - 2\psi_j + \psi_{j-1})/(\Delta x)^2\\ \delta_{yy}\psi_k &= (\psi_{k+1} - 2\psi_k + \psi_{k-1})/(\Delta y)^2\end{aligned}\right\} \tag{2}$$

The factored difference equations can be represented as

$$(I - \omega\delta_{xx})(I - \omega\delta_{yy})(\psi^{n+1} - \psi^n) = \omega(\delta_{xx} + \delta_{yy})\psi^n \tag{3}$$

or in algorithm form as

$$(I - \omega\delta_{xx})\Delta\psi^* = \omega(\delta_{xx} + \delta_{yy})\psi^n \tag{4a}$$

$$(I - \omega\delta_{yy})(\psi^{n+1} - \psi^n) = \Delta\psi^* \tag{4b}$$

One can flag off points by simply resetting ω as

$$\omega = \tilde{\omega}\mu_{j,k} \qquad \mu_{j,k} = \begin{cases}0 & \text{flag off}\\ 1 & \text{flag on}\end{cases} \tag{5}$$

Alternately one can initially proceed as if none of the points is special. Then, just before solving the first tridiagonal matrix corresponding to Eq. (4a), set to zero the appropriate elements of the matrix and its right-hand side. Consider the simple 6 × 6 system

$$\begin{bmatrix} b & c & & & & \\ a & b & c & & & \\ & a & b & c & & \\ & & a & b & c & \\ & & & a & b & c \\ & & & & a & b \end{bmatrix} \begin{bmatrix} \Delta\psi_1 \\ \Delta\psi_2 \\ \Delta\psi_3 \\ \Delta\psi_4 \\ \Delta\psi_5 \\ \Delta\psi_6 \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \end{bmatrix} \tag{6}$$

If elements in rows 3 and 4 are to be blanked, they are simply reset as

$$\begin{bmatrix} b & c & & & & \\ a & b & c & & & \\ & & 1 & & & \\ & & & 1 & & \\ & & & a & b & c \\ & & & & a & b \end{bmatrix} \begin{bmatrix} \Delta\psi_1 \\ \Delta\psi_2 \\ \Delta\psi_3 \\ \Delta\psi_4 \\ \Delta\psi_5 \\ \Delta\psi_6 \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ 0 \\ 0 \\ r_5 \\ r_6 \end{bmatrix} \tag{7}$$

In this example, nondiagonal elements in rows 3 and 4 are zeroed so that no change is computed for $(\psi^{n+1} - \psi^n)$. A similar treatment must occur in solving the tridiagonal matrix corresponding to Eq. (4b). Ultimately, the values of ψ at points 3 and 4 are updated by interpolation of values from another grid.

Although the above illustration is for a Laplacian, the Beam-Warming (9) algorithm for the Euler equations is treated in precisely the same manner. Overset grid results obtained using the Euler equation algorithm will be given elsewhere.

Finally, the boundary condition must be recoded. Interpolation routines to update overset grid boundaries must be provided, and modular coding is needed to account for the possibility that a minor grid may have different boundary-condition treatment than the major grid. For example, in Fig. 2 the $k = 1$ line in the major grid represents a periodicity boundary, whereas the $k = 1$ line for the minor nose grid represents a solid body boundary. Thus, separate boundary-condition routines must usually be supplied for each grid, and these should be provisioned for in a modular way.

LOCATING SPECIAL GRID POINTS

As part of the overall grid-generation package for the chimera grid scheme, a program must be included that locates and flags those points that must be blanked in the algorithm or that serve as special boundary points. To date, we have used simple bookkeeping procedures and allow the minor grids to interact with only the major grid. The minor grids are not allowed to interact with another body boundary (as shown in Fig. 3, for example). In time, these algorithms are expected to be refined, generalized, and made more efficient than current versions.

Blanking Points Within a Body Boundary

As indicated by Figs. 1-4, some of the points in the main grid will fall within the body boundary of a minor grid. These points must be blanked out. Because the minor grid points near the body boundary may be more finely spaced than the major grid points that are being turned off, we should blank additional major grid points until the grid spacings are more compatible. Thus, for the situation shown in Fig. 5, we may choose to turn off all major grid points within some boundary defined by the minor grid indices k = constant and $j = 1$ to JMAX. To achieve this we have used the following method.

A boundary curve corresponding to k = constant is defined within which points are to be blanked. Call this curve C (see Fig. 5). The origin to this region is defined by averaging all the points on C. We then search for the point on C which is the farthest from the origin. The distance from the origin to this point defines a radius, Rmax. The radius and angle for every point on curve C with respect to the origin are also computed and stored.

Points of the major grid are tested to determine if they fall within curve C. Provision can be made to exclude points that obviously cannot be within the closed-curve C by letting the user input the index bounds of test points. This increases the possibility for error, however, so such an option has not been provided. The first test, then, is to compute the distance between the major grid point being checked and the origin. If this distance exceeds Rmax then the point must lie outside of C; if the point is less than Rmax, its angle with respect to the origin is computed (see Fig. 6). The points along C are then searched to find that point on C which is closest in angle measure to the major grid point being tested. Points on C to either side are then tested so as to find angle bounds. (Because C is closed, the angle will jump from $-\pi$ to $+\pi$ somewhere between two points of C; if the test point has an angle within this range, the bounding C-curve points are already known.) If the radius from the origin to the test point is less than the weighted average of the radii of the bounding angles, the point is within C. A flag is then set and this information is stored. The above test is fairly reliable unless the body is concave (Fig. 7). In this case we can partition C into a C' and C" and work with two origins.

The previously described algorithm for locating points within a given boundary is only one of many that can be envisioned. An algorithm that can be more readily extended to three dimensions is sketched, although it has not been tested in computation. Let the boundary curve C be represented by k = constant as before and again assume that C is continuous and closed. If C corresponds to η = constant, and if the j index increases clockwise (see Fig. 8), then $\vec{\nabla}\eta$ represents an outward normal to C. Let Rp be the vector from the nearest point on C to the major grid point being tested. If the dot product of $\vec{\nabla}\eta$ with Rp is positive, the point being tested is outside of the curve C; if the dot product is negative, the point is inside of C.

Location and Interpolation of Mesh Overset Boundary Points

Outer boundary data for any minor grid must be provided by interpolating the solution of the major grid. If B is a point on the outer boundary of a minor grid, our first task is to locate the nearest major grid point (or points) from which interpolation can take place. The simplest test is to compute the distance between point B and points of the major grid and to select that major grid point which is closest to B. Again this search can be speeded up by excluding major grid points a large distance away.

If the boundary curve is smooth, one can also use a "stencil-walk" to speed the search. As sketched in

Fig. 9, after the nearest point to B is found, the nearest point to the next boundary point B' can be located by searching the stencil for the point that is nearest to B. Of the stencil of nine points, find that one which is nearest to B'. For this nearest point, search its stencil for the point closest to B', and so on, until the point tested is closer to B' than any of its stencil neighbors.

The major grid will have various "holes" of blanked points. The perimeter of these points serves as an inner boundary to the major grid, and solution data must be supplied to this perimeter. In this case the nearest minor grid point must be located for interpolation of minor grid data onto the perimeter of the blanked major grid. A search similar to that of the first approach outlined is used.

For interpolating the grid boundary data from the field of another grid we have simply used second-order Taylor series approximations of the form

$$\psi = \psi_0 + \Delta x\psi_x + \Delta y\psi_y + \frac{(\Delta x)^2}{2}\psi_{xx} + \Delta x\Delta y\psi_{xy} + \frac{(\Delta y)^2}{2}\psi_{yy} \qquad (8)$$

where

$$\psi_x = \xi_x\psi_\xi + \eta_x\psi_\eta \text{ (etc.)}$$

and Δx and Δy are the increments between the point from which the interpolation is being made and the point being interpolated for. A linear combination of such interpolation formulas can be written from nearby points to ensure a very smooth interpolation. We have not needed such a technique to date.

Because the search of nearest points is limited to the index range that excludes fixed boundaries, the derivatives of ψ_x and ψ_{xx}, etc., can always be formed with central differencing. This simplifies the interpolation formula insofar that it does not require the use of special one-sided differencing at boundaries.

A criticism of using such a simple interpolation formula is that conservation is not maintained. That is, although we may difference the flow equations in divergence form, the divergence property is not numerically preserved if some values are obtained by interpolation. This may be a problem if a jump discontinuity such as a shock wave crosses the interpolation boundary. Perhaps in this case interpolated values can be later adjusted to satisfy a numerical line-integral of flux.

RESULTS

As part of a package for generating overset grids, a flow-solver code has been developed which solves the incompressible stream function. Although such a program is useful in its own right, the application here is to uncover logic errors in locating and flagging special points in various kinds of overset mesh systems. The ultimate application for this linear code will be to use it to assess the quality of a given generated overset mesh system. For example, the Laplacian can be economically solved to help determine whether an adequate number of points are provided at, say, an airfoil nose region or whether an additional grid should be introduced to resolve some other feature. Specifically, Eq. (1) in generalized coordinates is solved using the algorithm given by Eq. (4). Blanked points are treated by using the relaxation factor as given by Eq. (5).

As a first test of the overset grid system, a body-fitted O-grid was superimposed on a stretched rectangular grid which served as the major grid. Figure 10a shows an overview of an O-grid about an NACA 0012 airfoil overset on the rectangular grid; a closeup is shown in Fig. 10b. The circular symbols plotted over points of the rectangular grid indicate nearest major grid points that are used for interpolating the outer boundary points of the O-grid. Here the O-grid is considered to be the minor grid. The plotted x symbols indicate points in the rectangular grid that are blanked out. Although not specially flagged, the perimeter of the blanked-out points is updated by interpolating nearby minor grid values. In this case all rectangular points within the $k = 3$ grid line of the O-grid are blanked. A perimeter of rectangular points adjacent to those blanked out is also flagged. The calculated pressure distribution is indicated in Fig. 10c for incompressible flow and is shown compared with exact theory; the comparison is satisfactory.

The grids shown in Figs. 11a and 11b show a 12% ellipse modeling a nonlifting biplane configuration. The lower boundary of the rectangular mesh serves as a line of symmetry. Unlike the previous example, this geometry would be poorly meshed by use of only a single O-grid. This is because the line of symmetry is so close that the O-grid would become very distorted. The computed pressure coefficient is indicated in Fig. 11c.

Another application of the overset mesh system is indicated by Figs. 12 and 13 for an inlet with and without a centerbody. In the first case (Fig. 12), a stretched rectangular mesh is used as the major grid and a body-fitted C-grid is used to resolve the inlet. The detailed grid view (Fig. 12a) shows blanked-out points and nearby interpolation points on the rectangular grid. Nearby points on the minor body-fitted grid which are used to interpolate the perimeter of blanked-out rectangular points are not shown. By choosing various values of stream function to be constant on the body boundary we can control the mass flow into the inlet. Computed streamlines are shown in Figs. 12b and 12c for mass flow rates that cause spillage (Fig. 12b) and suction (Fig. 12c). An additional set of grids and computational results are shown in Figs. 13a-13c. In this latter case the inlet has a centerbody that is fitted by shearing the major rectangular grid over this boundary.

A final example shows an airfoil and flap arrangement using O-grids for both the major and minor grids. This arrangement, in which the flap is tucked in below the main airfoil, cannot be nicely fitted by a single mapping with cuts. Overviews of the grids and computed results are indicated in Figs. 14a-14d. The circulation of each profile is found automatically in these cases by requiring a match of trailing-edge pressures as a Kutta condition. Using the Bernoulli equation, the condition of setting the pressures at the point above the trailing edge, u, to the point below, l, gives the relation

$$[(\eta_x^2 + \eta_y^2)\psi_\eta^2]_{upper} = [(\eta_x^2 + \eta_y^2)\psi_\eta^2]_{lower} \qquad (9)$$

This relation uses the inviscid tangency-boundary condition that is constant on the body so $\psi_\xi = 0$; ξ and η

represent the curvilinear body-conforming coordinates with ξ circumferential and η directed outward. Differencing this relation at the trailing edge results in a solution for the stream function on the body in terms of neighboring ψ-values at the trailing edge.

CONCLUDING REMARKS

It has been known for a long time that multiple-grid systems work and that they offer computational advantages. Although using multiple grid systems is conceptually straightforward, many experienced practitioners of computational fluid dynamics (CFD) who have coded such programs would likely agree that all of the extra bookkeeping, although simple, tends to be tedious. It is clear, however, that some type of multiple grid will be utilized. A major task in CFD, then, will be to design computer codes that can use multiple grids and still remain usable and readable.

In this paper we have described a study of a chimera grid code architecture using overset grids. We are finding that such a grid system need not overly complicate the computer code, that it can be handled within a general framework, and that it can simplify the task of grid generation.

ACKNOWLEDGMENT

The authors are indebted to Pieter Buning, Ames Research Center, who gave valuable suggestions and provided the grid plotting routines.

REFERENCES

1. Magnus, R. and Yoshihara, H., "Inviscid Transonic Flow over Airfoils," AIAA Paper 70-47, 1970.

2. Berger, M. and Oliger, J., "Adaptive Methods for Hyperbolic Partial Differential Equations," Numerical Analysis Project Manuscript N.A. 83-02, Computer Science Dept., Stanford, California 94305.

3. Starius, G., "On Composite Mesh Difference Methods for Hyperbolic Differential Equations," Numerische Mathematik, Vol. 35, Springer-Verlag, 1980.

4. Atta, E. H., "Component Adaptive Grid Interfacing," AIAA Paper 81-0382, 1981.

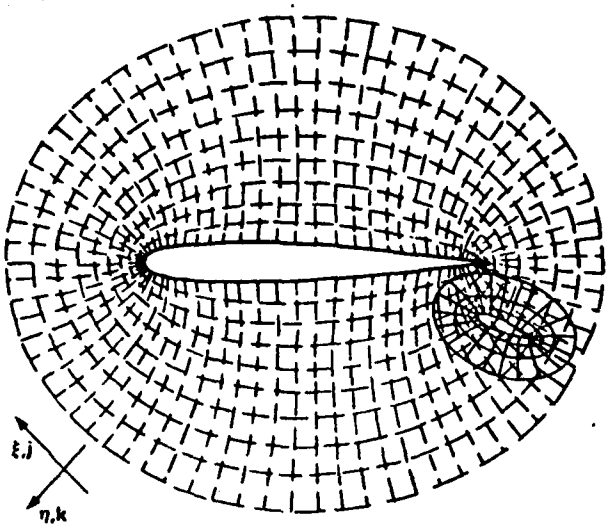
5. Atta, E. H. and Vadyak, J. A., "Grid Interfacing Zonal Algorithm for Three-Dimensional Transonic Flows about Aircraft Configurations," AIAA Paper 82-1017, 1982.

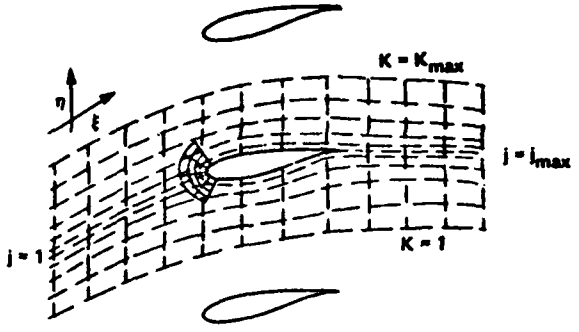
6. Steger, J. L., "Implicit Finite Difference Simulation of Inviscid and Viscous Compressible Flow," Transonic, Shock and Multidimensional Flows, R. E. Meyers, ed., Academic Press, New York, 1982.

7. Steger, J. L., "On Application of Body-Conforming Curvilinear Grids for Finite Difference Solution of External Flow," Numerical Grid Generation, J. F. Thompson, ed., North-Holland, New York, 1982.

8. Steger, J. L. and Lomax, H., "Calculation of Inviscid Shear Flow Using a Relaxation Method for the Euler Equations, NASA SP-347, 1975.

9. Beam, R. and Warming, R. F., "An Implicit Finite-Difference Algorithm for Hyperbolic Systems in Conservation-Law-Form," Journal of Computational Physics, Vol. 22, 1976.





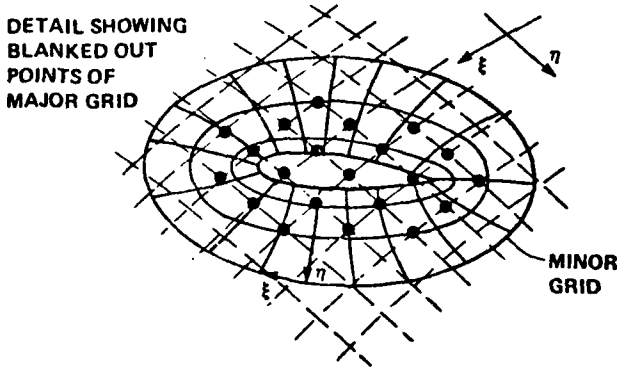


Fig. 1 Overset grids for airfoil with flap

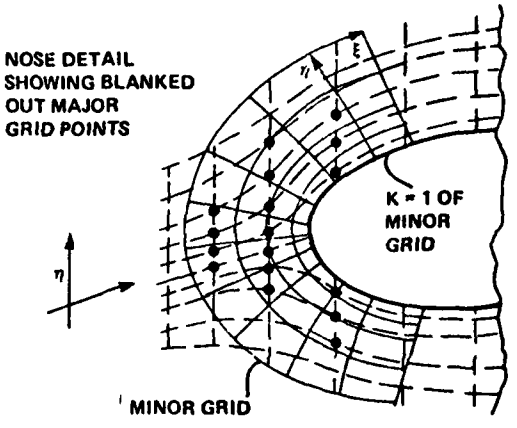


Fig. 2 Overset grids for cascade blades with minor grid used to resolve nose of blade

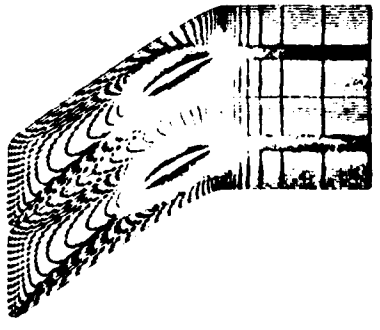


Fig. 3 C-grid for cascade, showing extensive skewness

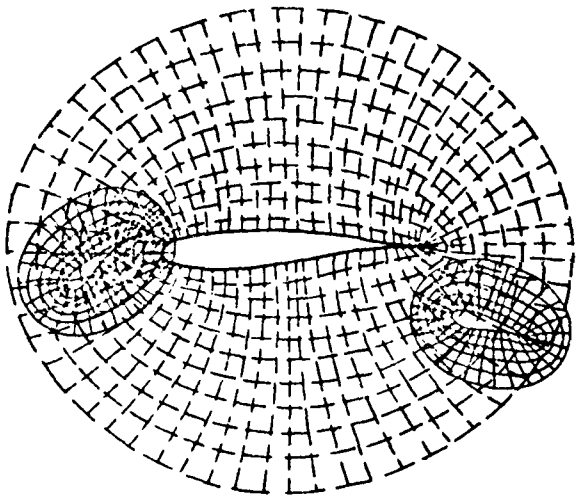


Fig. 4 Airfoil with leading- and trailing-edge flaps resolved with overset grids; note that the grid spacings are not shown to correct scale

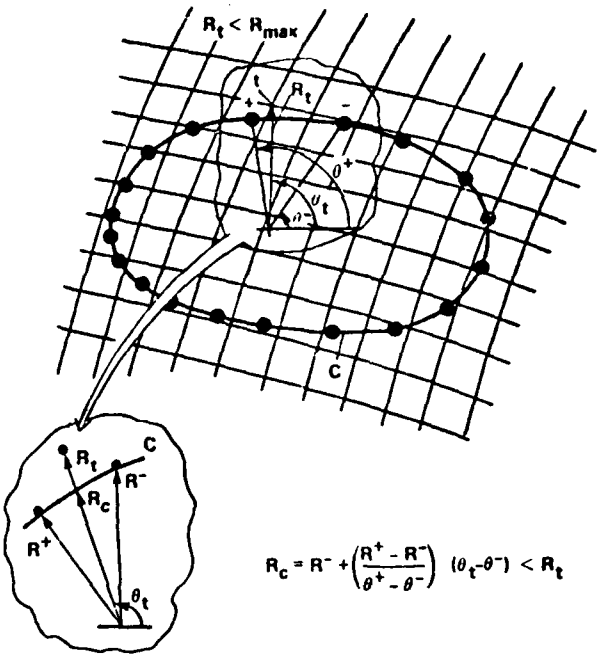


Fig. 6 Test procedure for locating major grid points that lie within curve C

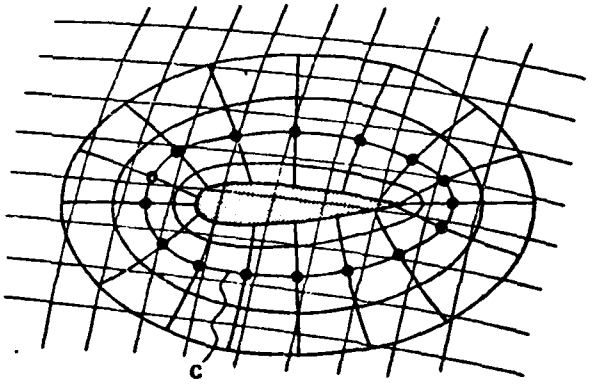


Fig. 5 Curve C which defines region of blanked-out major grid points

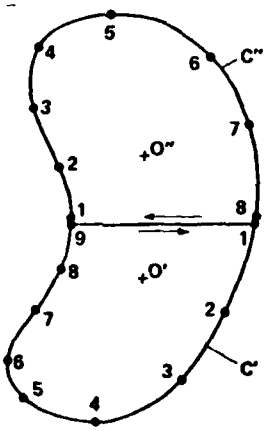


Fig. 7 Partition of curve C for case of concave boundary

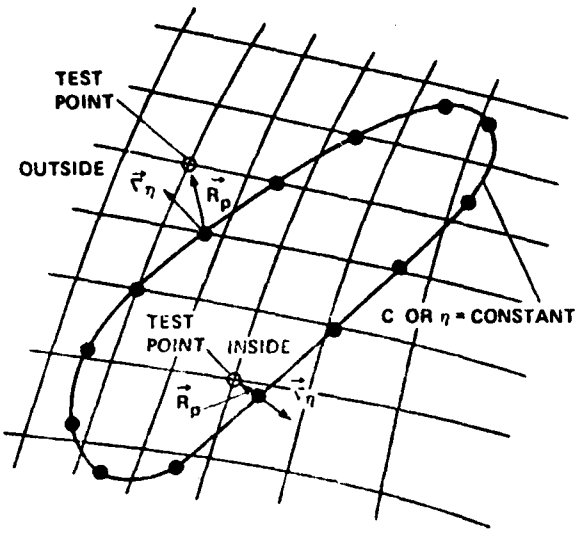


Fig. 8 Normal vector search procedure for locating points within curve C

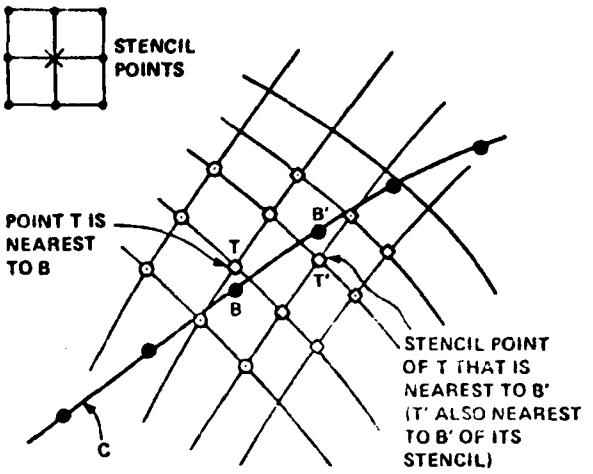
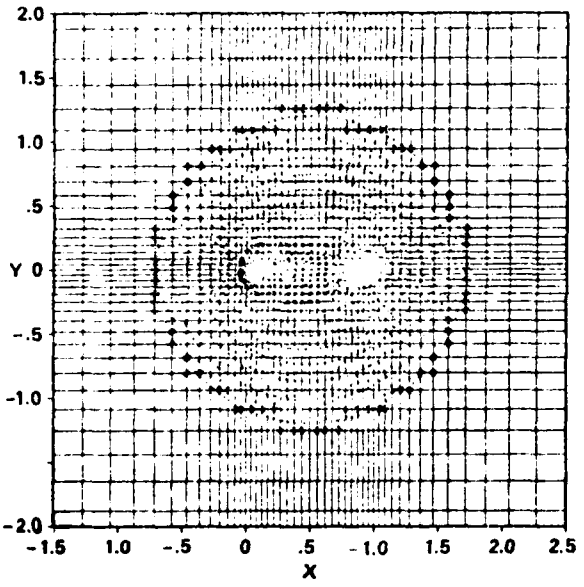
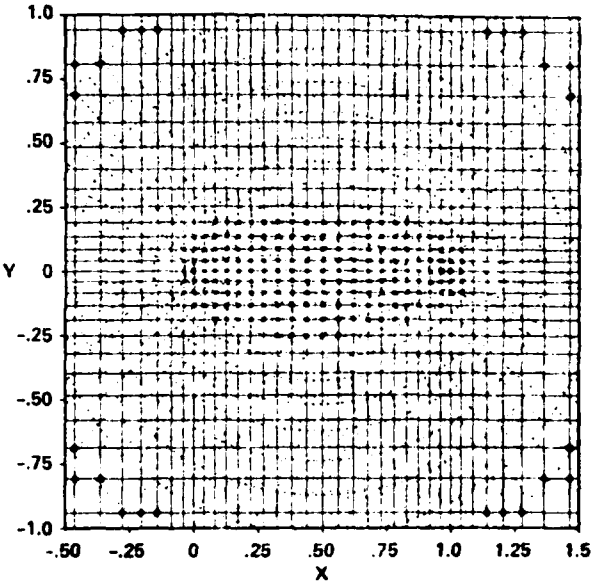


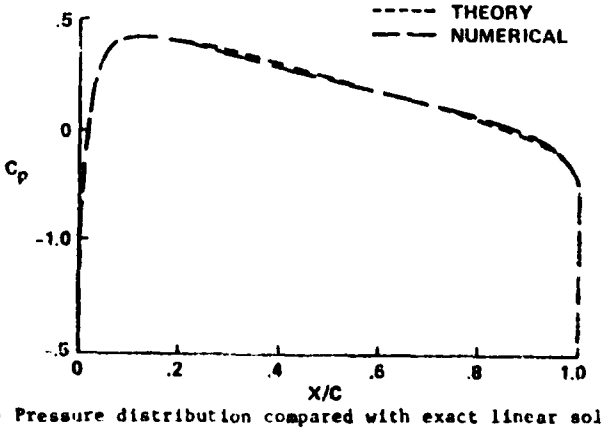
Fig. 9 Stencil walk from point T to the point T' which is nearest to B'



a) Overview

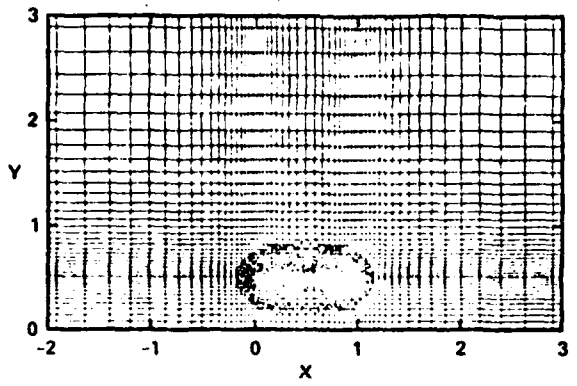


b) Detailed view

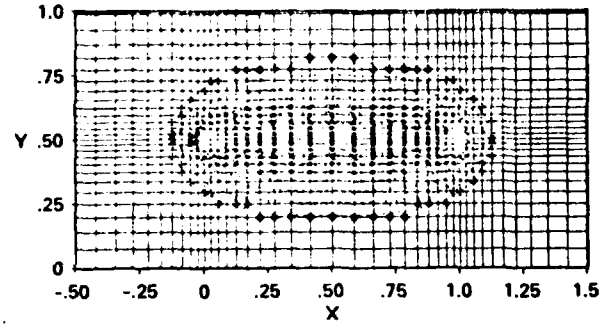


c) Pressure distribution compared with exact linear solution

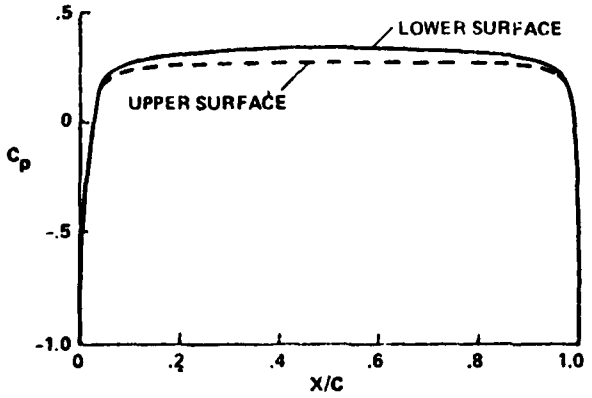
Fig. 10 O-grid about NACA 0012 airfoil on Cartesian major grid.



a) Overset grids used to revolve wing very close to plane of symmetry

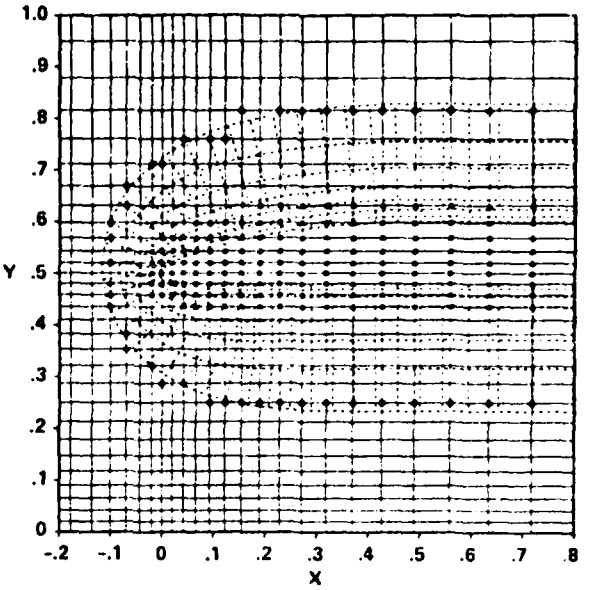


b) Overview showing blanked-out and interpolated points

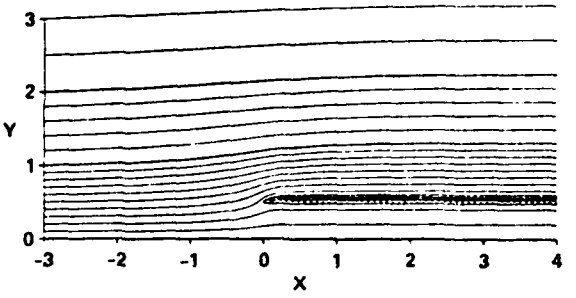


c) Pressure distribution

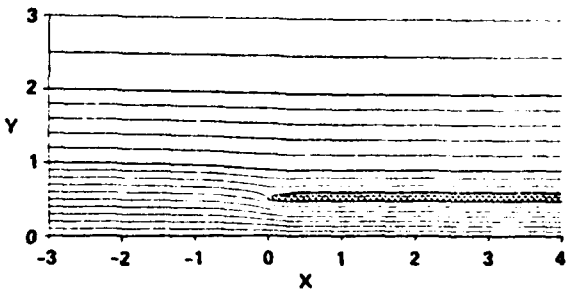
Fig. 11 Ellipse (12%) modeling a nonlifting biplane configuration



a) Close-up view of inlet grids showing lip detail

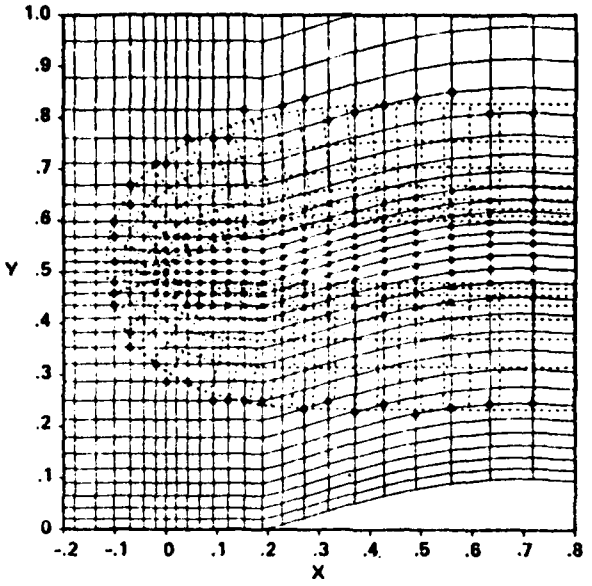


b) Streamlines with spillage

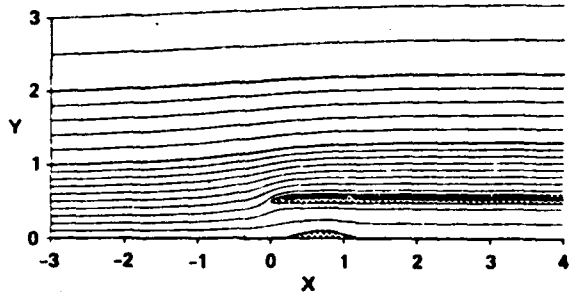


c) Streamlines with suction

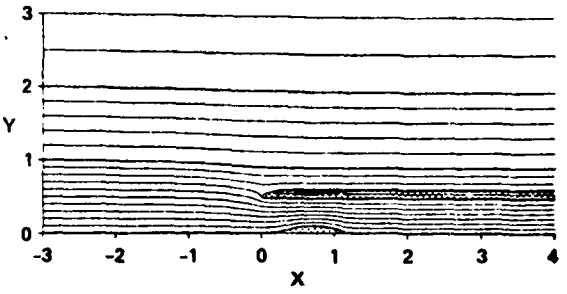
Fig. 12 Inlet without centerbody



a) Close-up view

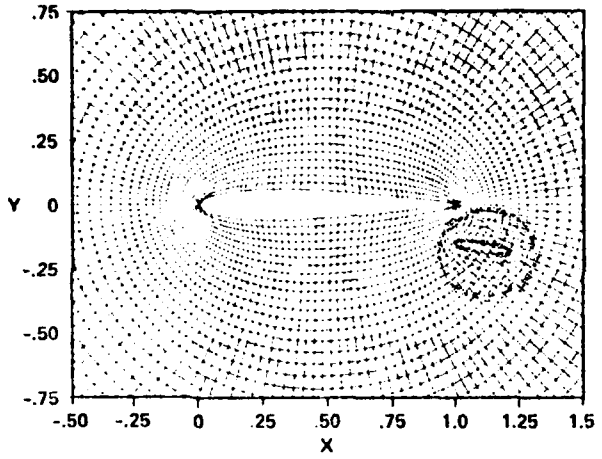


b) Streamlines with spillage

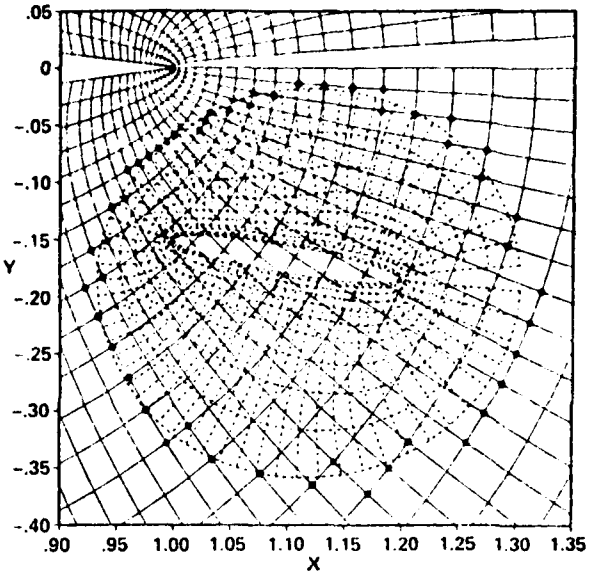


c) Streamlines with suction

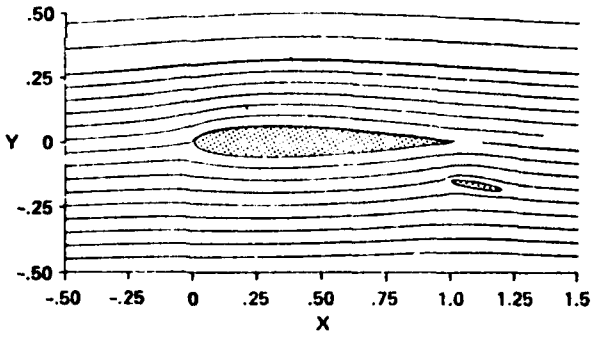
Fig. 13 Inlet with centerbody



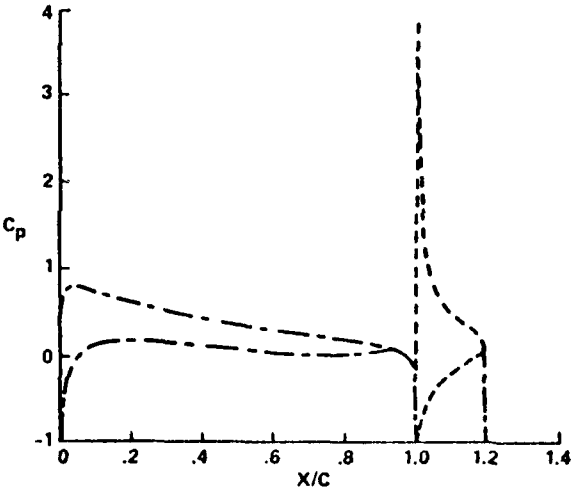
a) Use of overset O-grids



b) Close-up of flap grid



c) Streamlines



d) Incompressible pressure distributions

Fig. 14 Airfoil and flap arrangement

APPENDIX D

A CONSERVATIVE FINITE DIFFERENCE ALGORITHM FOR THE UNSTEADY TRANSONIC POTENTIAL EQUATION IN GENERALIZED COORDINATES

J. O. Bridgeman
NASA-Ames Research Center
Moffett Field, CA

J. L. Steger
Stanford University
Stanford, CA

F. X. Caradonna
U.S. Army Research and
Technology Laboratories (AVRADCOM)
Ames Research Center
Moffett Field, CA

AIAA 9th Atmospheric Flight
Mechanics Conference
August 9-11, 1982, San Diego, California

A CONSERVATIVE FINITE DIFFERENCE ALGORITHM FOR THE UNSTEADY TRANSONIC POTENTIAL EQUATION IN GENERALIZED COORDINATES

J. O. Bridgeman*
NASA Ames Research Center, Moffett Field, California

J. L. Steger†
Department of Aeronautics and Astronautics
Stanford University, Stanford, California

and

F. X. Caradonna*
Army Aeromechanics Laboratory, AVRADCOM
Ames Research Center, Moffett Field, California

I. Abstract

An implicit, approximate-factorization, finite-difference algorithm has been developed for the computation of unsteady, inviscid transonic flows in two and three dimensions. The computer program solves the full-potential equation in generalized coordinates in conservation-law form in order to properly capture shock-wave position and speed. A body-fitted coordinate system is employed for the simple and accurate treatment of boundary conditions on the body surface. The time-accurate algorithm is modified to a conventional ADI relaxation scheme for steady-state computations. Results from two- and three-dimensional steady and two-dimensional unsteady calculations are compared with existing methods.

II. Introduction

Modern transport, military and rotor aircraft routinely operate in the transonic flight regime. In transonic flight, the aerodynamic forces are extremely sensitive to small perturbations in the motion of the vehicle. Unsteady flow adjustment is very slow as upstream propagation of information is restricted by locally high subsonic or supersonic flow resulting in large phase lags. In addition, the existence of embedded shock waves with the potential for large excursions further complicates the transonic-flow environment. Thus, it is not surprising that the most critical aeroelastic phenomena occur in this flight regime. And while the supercritical airfoil sections are more aerodynamically efficient than conventional airfoils, they are more susceptible to flutter.[1] Linearized, unsteady subsonic aerodynamic theory is incapable of predicting these complicated flows. In addition, transonic prediction methods that use the harmonic approach to compute a small perturbation from nonlinear, mean steady-state flow have limited applicability. They are valid only for small-amplitude, high-frequency flows and cannot account for shock-wave motions. Thus, it is of paramount importance to develop methods that can properly compute shock waves and their motions if their role in transonic, aeroelastic stability is to be accounted for.[2]

Unsteady, transonic, aerodynamic methods based on various nonlinear small-disturbance potential equations have been developed by several researchers. Ballhaus and Steger[3] and Ballhaus and Goorjian[4] have developed an efficient method for solving the two-dimensional low-frequency transonic small-disturbance equation. The resultant code, LTRAN2, has been extensively used in spite of its various limitations. Recently, methods that moderately extend the frequency and Mach number range of this equation have been developed.[5-7] In addition, the effect of viscous corrections has been examined by Rizzetta.[8] In another effort, Borland, Rizzetta, and Yoshihara[9] have recently reported on an algorithm for a modified form of the small-disturbance equation in three dimensions with no frequency limitations. This code is currently being used to compute unsteady aerodynamics for transonic flutter analyses.[10,11]

A practical tool for the computation of unsteady, inviscid transonic flow about complex configurations must accurately simulate the significant physics and be computationally efficient for routine use in engineering analysis. Unsteady, full-potential theory can satisfactorily replace Euler equation solutions if the shock waves are sufficiently weak and yet maintain computer time and storage requirements similar to the simplified small-disturbance theory. The unsteady, full-potential equation has been solved using fully implicit methods by Goorjian,[12] Steger and Caradonna,[13] Chipman and Jameson,[14] and Sankar et al.[15,16] The Chipman and Jameson procedure solves a system of two equations for the two unknowns, density and velocity potential, using an approximate-factorization scheme. The method of Goorjian uses an ADI scheme to solve a scalar equation for the velocity potential by employing a time-linearization of the density similar to the present approach. To date both of these methods have only been applied to two-dimensional nonlifting pulsating airfoils using a simple, sheared mesh. The method of Sankar et al. uses the strongly implicit procedure and has been applied to compute steady and unsteady flow over wings using a sheared, parabolic coordinate system.

The present procedure is an extension of the Steger and Caradonna work[13] which used a stretched Cartesian grid with small-disturbance planar boundary conditions. Currently, a generalized body-fitted coordinate system is employed that can be adapted to wings, wing-body combinations, and bodies of revolution. This method is unique as it can use the mesh-point-efficient spherical grids that are presently being developed in conjunction with this code. Steger and Caradonna reported that the unsteady scheme was a very poor relaxation

---

*Research Scientist. Member AIAA.
†Associate Professor. Member AIAA.

This paper is declared a work of the U.S. Government and therefore is in the public domain

algorithm. In this work, the scheme is modified to an ADI relaxation procedure for computing fast, steady-state solutions by a simple flag.

The governing equations and boundary conditions are discussed in Section II. In Section III the conservative differencing and numerical algorithm for the solution of the finite-difference equations are presented. In Section IV, results are presented for various two- and three-dimensional cases to verify the algorithm. Finally, concluding remarks are made in Section V.

III. Mathematical Formulation

Full-Potential Equation

The three-dimensional, unsteady, full-potential equation in strong conservation-law form is given by

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x}(\rho\phi_x) + \frac{\partial}{\partial y}(\rho\phi_y) + \frac{\partial}{\partial z}(\rho\phi_z) = 0 \qquad (1)$$

where the density ρ is determined from the unsteady Bernoulli relation, for steady, uniform incoming flow

$$\rho = \left[1 + \frac{\gamma - 1}{2}(M_\infty^2 - 2\phi_t - \phi_x^2 - \phi_y^2 - \phi_z^2)\right]^{1/\gamma - 1} \qquad (2)$$

The density ρ and velocity components ϕ_x, ϕ_y, and ϕ_z are nondimensionalized by the free-stream ρ_∞ and speed of sound a_∞. The Cartesian coordinates x, y, and z are nondimensionalized by a reference length ℓ, such as the airfoil chord c, and the time t is referenced to a_∞/ℓ.

Equations (1) and (2) express mass conservation for unsteady, inviscid, isentropic, and irrotational flow. The corresponding shock-jump conditions can be a suitable approximation to the Rankine-Hugoniot relations for many transonic flow applications if the shock waves are sufficiently weak. In deriving Eq. (2) the far-field is assumed to be steady.

Transformation of the Equations

The treatment of arbitrary body boundaries is usually made more convenient by the use of a coordinate transformation which maps the body surface to a rectangular coordinate surface in the transformed plane. Boundary conditions at the body surface can then be simply and accurately treated. In addition, these mappings can be used to cluster grid points in regions of the flow with high gradients, thereby enhancing numerical solution accuracy. A general independent variable transformation is indicated by

$$\xi = \xi(x,y,z,t)$$

$$\eta = \eta(x,y,z,t)$$

$$\zeta = \zeta(x,y,z,t)$$

$$\tau = t$$

The strong conservation-law form of Eq. (1) is maintained by expressing it as

$$\frac{\partial}{\partial \tau}\left(\frac{\rho}{J}\right) + \frac{\partial}{\partial \xi}\left(\frac{\rho U}{J}\right) + \frac{\partial}{\partial \eta}\left(\frac{\rho V}{J}\right) + \frac{\partial}{\partial \zeta}\left(\frac{\rho W}{J}\right) = 0 \qquad (3)$$

with the Bernoulli relation, Eq. (2), transforming as

$$\rho = \left\{1 + \frac{\gamma - 1}{2}\left[M_\infty^2 - 2\phi_\tau - (U + \xi_t)\phi_\xi - (V + \eta_t)\phi_\eta - (W + \zeta_t)\phi_\zeta\right]\right\}^{1/\gamma - 1}$$

where

$$\left.\begin{aligned} U &= \xi_t + A_1\phi_\xi + A_4\phi_\eta + A_5\phi_\zeta \\ V &= \eta_t + A_4\phi_\xi + A_2\phi_\eta + A_6\phi_\zeta \\ W &= \zeta_t + A_5\phi_\xi + A_6\phi_\eta + A_3\phi_\zeta \end{aligned}\right\} \qquad (5)$$

and

$$\left.\begin{aligned} A_1 &= \xi_x^2 + \xi_y^2 + \xi_z^2 \\ A_2 &= \eta_x^2 + \eta_y^2 + \eta_z^2 \\ A_3 &= \zeta_x^2 + \zeta_y^2 + \zeta_z^2 \\ A_4 &= \xi_x^2\eta_x^2 + \xi_y^2\eta_y^2 + \xi_z^2\eta_z^2 \\ A_5 &= \xi_x^2\zeta_x^2 + \xi_y^2\zeta_y^2 + \xi_z^2\zeta_z^2 \\ A_6 &= \eta_x^2\zeta_x^2 + \eta_y^2\zeta_y^2 + \eta_z^2\zeta_z^2 \end{aligned}\right\} \qquad (6a)$$

$$J = \xi_x(\eta_y\zeta_z - \eta_z\zeta_y) + \eta_x(\xi_z\zeta_y - \xi_y\zeta_z) + \zeta_x(\xi_y\eta_z - \xi_y\eta_y) \qquad (6b)$$

Here, U, V, and W are contravariant velocities along the ξ, η, and ζ directions, respectively, $A_1 - A_6$ are metric quantities, and J is the Jacobian of the transformation $|\partial(\xi,\eta,\zeta)/\partial(x,y,z)|$. The metric quantities in Eqs. (6a) and (6b) are evaluated using the following metric identities

$$\left.\begin{aligned} \xi_x &= J(y_\eta z_\zeta - y_\zeta z_\eta) & \zeta_x &= J(y_\xi z_\eta - y_\eta z_\xi) \\ \xi_y &= J(x_\zeta z_\eta - x_\eta z_\zeta) & \zeta_y &= J(x_\eta z_\xi - x_\xi z_\eta) \\ \xi_z &= J(x_\eta y_\zeta - x_\zeta y_\eta) & \zeta_z &= J(x_\xi y_\eta - x_\eta y_\xi) \\ \eta_x &= J(y_\zeta z_\xi - y_\xi z_\zeta) & \xi_t &= -x_\tau\xi_x - y_\tau\xi_y - z_\tau\xi_z \\ \eta_y &= J(x_\xi z_\zeta - x_\zeta z_\xi) & \eta_t &= -x_\tau\eta_x - y_\tau\eta_y - z_\tau\eta_z \\ \eta_z &= J(x_\zeta y_\xi - x_\xi y_\zeta) & \zeta_t &= -x_\tau\zeta_x - y_\tau\zeta_y - z_\tau\zeta_z \end{aligned}\right\} \qquad (7)$$

Boundary Conditions

At the body surface, flow tangency is required (i.e., no flow through the body). This imposed by setting the contravariant velocity in the ζ direction to zero, viz

$$W = \zeta_t + A_5\phi_\xi + A_6\phi_\eta + A_3\phi_\zeta = 0 \qquad (8)$$

A similar condition (i.e., $V = 0$) is imposed at the symmetry plane for wing cases.

For lifting cases, allowance must be made for a jump of potential across a wake-like cut. In unsteady flow, vorticity is continuously shed from the wing. In a potential formulation this is approximately modeled with a cut aligned with the

shear layer. Here we further idealize this flow by keeping the cut in the mean chord line plane. By imposing the usual shear-layer assumptions, an equation for the jump in potential,

$$\Gamma = [\phi] = \phi_u - \phi_\ell$$

along the cut can be derived. Across an inviscid shear layer, normal velocity and pressure are continuous. Since isentropic flow has been assumed, the density is also taken to be continuous. The Bernoulli relation together with the continuity of density requires that

$$\Gamma_\tau + \langle V\rangle\Gamma_\eta + \langle W\rangle\Gamma_\zeta = 0 \qquad (9)$$

where $\langle V\rangle$ and $\langle W\rangle$ are the averages of the contravariant velocities above and below the wake, i.e.,

$$\langle V\rangle = \frac{1}{2}(V_u + V_\ell)$$

$$\langle W\rangle = \frac{1}{2}(W_u + W_\ell)$$

This convection equation for Γ is imposed in the wake.

At distances far from the body the flow is required to be free stream, which, with the use of ρ_∞ and a_∞ as the references, is

$$\phi = M_\infty x$$

$$\rho = 1$$

where M_∞ is the free-stream Mach number. For steady lifting cases, the velocity potential at the outer boundary is updated with the usual compressible vortex solution with strength Γ.

III. Numerical Algorithm

Temporal Differencing

A first-order-time-accurate approximation to Eq. (3) is obtained using Euler backward differencing

$$\hat\rho^{n+1} - \hat\rho^n + h[\partial_\xi(\hat\rho U)^{n+1} + \partial_\eta(\hat\rho V)^{n+1} + \partial_\zeta(\hat\rho W)^{n+1}] = 0 \qquad (10)$$

where $h = \Delta t$ and $\hat\rho = \rho/J$. To solve a scalar system of equations for the potential at each new time level the density is linearized about old time levels. The density coefficients in the spatial derivative terms are lagged to level n. A linearization resulting in a conservative differencing of the time derivative of the density is obtained by noting that $\rho = \rho(\phi)$ and using a Taylor series expansion

$$\hat\rho^{n+1} - \hat\rho^n = \left[\rho^n + \left(\frac{1}{J}\frac{\partial\rho}{\partial\phi}\right)^n(\phi^{n+1} - \phi^n)\right] - \left[\hat\rho^{n-1} + \left(\frac{1}{J}\frac{\partial\rho}{\partial\phi}\right)^{n-1}(\phi^n - \phi^{n-1})\right] + O(h^2) \qquad (11)$$

where $\partial\rho/\partial\phi$ is a differential operator obtained from the Bernoulli relation

$$\frac{\partial\rho}{\partial\phi} = -\rho^{2-\gamma}[\partial_\tau + U\partial_\xi + V\partial_\eta + W\partial_\zeta] \qquad (12)$$

Finally, to better facilitate the application of approximate factorization techniques, the cross-derivative terms are lagged in time by rewriting the implicit spatial derivatives in the form

$$\left.\begin{aligned}\partial_\xi(\hat\rho U)^{n+1} &= \partial_\xi(\hat\rho A_1)^n\partial_\xi(\phi^{n+1} - \phi^n) + \partial_\xi(\hat\rho U)^n + O(h)\\ \partial_\eta(\hat\rho V)^{n+1} &= \partial_\eta(\hat\rho A_2)^n\partial_\eta(\phi^{n+1} - \phi^n) + \partial_\eta(\hat\rho V)^n + O(h)\\ \partial_\zeta(\hat\rho W)^{n+1} &= \partial_\zeta(\hat\rho A_3)^n\partial_\zeta(\phi^{n+1} - \phi^n) + \partial_\zeta(\hat\rho W)^n + O(h)\end{aligned}\right\} \qquad (13)$$

Combining Eqs. (10)-(13) yields the conservative time-discretized form of the full-potential equation involving only the potential at the new time level

$$\begin{aligned}&\Big\{I + h(U^n\partial_\xi + V^n\partial_\eta + W^n\partial_\zeta) - \frac{h^2}{\hat\beta^n}[\partial_\xi(\hat\rho A_1)^n\partial_\xi\\ &+ \partial_\eta(\hat\rho A_2)^n\partial_\eta + \partial_\zeta(\hat\rho A_3)^n\partial_\zeta]\Big\}(\phi^{n+1} - \phi^n)\\ &= (\phi^n - \phi^{n-1}) + \frac{\hat\beta^{n-1}}{\hat\beta^n}(\phi^n - 2\phi^{n-1} + \phi^{n-2})\\ &+ \frac{h}{\hat\beta^n}(\hat\rho^n - \hat\rho^{n-1}) + h\frac{\hat\beta^{n-1}}{\hat\beta^n}(U^{n-1}\partial_\xi + V^{n-1}\partial_\eta\\ &+ W^{n-1}\partial_\zeta)(\phi^n - \phi^{n-1}) + \frac{h^2}{\hat\beta^n}[\partial_\xi(\hat\rho U)^n + \partial_\eta(\hat\rho V)^n\\ &+ \partial_\zeta(\hat\rho W)^n]\end{aligned} \qquad (14)$$

where $\hat\beta = \rho^{2-\gamma}/J$.

Spatial Differencing

An optionally first- or second-order-accurate spatial differencing of Eq. (14) is given by (note: the spatial indices are suppressed here for convenience)

$$\begin{aligned}&\Big\{I + h(U^n\delta_\xi + V^n\delta_\eta + W^n\delta_\zeta) - \frac{h^2}{\hat\beta^n}[\bar\delta_\xi(\hat\rho A_1)^n\bar\delta_\xi\\ &+ \bar\delta_\eta(\hat\rho A_2)^n\bar\delta_\eta + \bar\delta_\zeta(\hat\rho A_3)^n\bar\delta_\zeta]\Big\}(\phi^{n+1} - \phi^n)\\ &= (\phi^n - \phi^{n-1}) + \frac{\hat\beta^{n-1}}{\hat\beta^n}(\phi^n - 2\phi^{n-1} + \phi^{n-2})\\ &+ \frac{h}{\hat\beta^n}(\hat\rho^n - \hat\rho^{n-1}) + h\frac{\hat\beta^{n-1}}{\hat\beta^n}[U^{n-1}\delta_\xi + V^{n-1}\delta_\eta\\ &+ W^{n-1}\delta_\zeta](\phi^n - \phi^{n-1}) + \frac{h^2}{\hat\beta^n}\Big[\bar\delta_\xi\left(\frac{\hat\rho\bar U}{J}\right)^n + \bar\delta_\eta(\hat\rho\bar V)^n\\ &+ \bar\delta_\zeta(\hat\rho\bar W)^n\Big]\end{aligned} \qquad (15)$$

where

$$\left.\begin{aligned}
\bar{\delta}_\xi(\hat{\rho}A_1)\bar{\delta}_\xi\phi_j &= \left[A_{1_{j+1/2}}J^{-1}_{j+1/2}\left(\frac{\rho_{j+1}+\rho_j}{2}\right)(\phi_{j+1}-\phi_j)\right.\\
&\quad \left. - A_{1_{j-1/2}}J^{-1}_{j-1/2}\left(\frac{\rho_j+\rho_{j-1}}{2}\right)(\phi_j-\phi_{j-1})\right]\\
\bar{\delta}_\eta(\hat{\rho}A_2)\bar{\delta}_\eta\phi_k &= \left[A_{2_{k+1/2}}J^{-1}_{k+1/2}\left(\frac{\rho_{k+1}+\rho_k}{2}\right)(\phi_{k+1}-\phi_k)\right.\\
&\quad \left. - A_{2_{k-1/2}}J^{-1}_{k-1/2}\left(\frac{\rho_k+\rho_{k-1}}{2}\right)(\phi_k-\phi_{k-1})\right]\\
\bar{\delta}_\zeta(\hat{\rho}A_3)\bar{\delta}_\zeta\phi_\ell &= \left[A_{3_{\ell+1/2}}J^{-1}_{\ell+1/2}\left(\frac{\rho_{\ell+1}+\rho_\ell}{2}\right)(\phi_{\ell+1}-\phi_\ell)\right.\\
&\quad \left. - A_{3_{\ell-1/2}}J^{-1}_{\ell-1/2}\left(\frac{\rho_\ell+\rho_{\ell-1}}{2}\right)(\phi_\ell-\phi_{\ell-1})\right]
\end{aligned}\right\} \tag{16}$$

and

$$\left.\begin{aligned}
\delta_\xi\left(\frac{\tilde{\rho}\bar{U}}{J}\right) &= J^{-1}_{j+1/2}\tilde{\rho}_{j+1/2}\bar{U}_{j+1/2} - J^{-1}_{j-1/2}\tilde{\rho}_{j-1/2}\bar{U}_{j-1/2}\\
\delta_\eta(\hat{\rho}\bar{V}) &= J^{-1}_{k+1/2}\left(\frac{\rho_{k+1}+\rho_k}{2}\right)\bar{V}_{k+1/2} - J^{-1}_{k-1/2}\left(\frac{\rho_k+\rho_{k-1}}{2}\right)\bar{V}_{k-1/2}\\
\delta_\zeta(\hat{\rho}\bar{W}) &= J^{-1}_{\ell+1/2}\left(\frac{\rho_{\ell+1}+\rho_\ell}{2}\right)\bar{W}_{\ell+1/2} - J^{-1}_{\ell-1/2}\left(\frac{\rho_\ell+\rho_{\ell-1}}{2}\right)\bar{W}_{\ell-1/2}
\end{aligned}\right\} \tag{17}$$

The contravariant velocities at the mesh half-points are determined from

$$\left.\begin{aligned}
\bar{U}_{j+1/2} &= \xi_{t_{j+1/2}} + A_{1_{j+1/2}}(\phi_{j+1}-\phi_j)\\
&\quad + \tfrac{1}{2}\left[A_{4_{j+1}}\delta_\eta\phi_{j+1} + A_{4_j}\delta_\eta\phi_j\right]\\
&\quad + \tfrac{1}{2}\left[A_{5_{j+1}}\delta_\zeta\phi_{j+1} + A_{5_j}\delta_\zeta\phi_j\right]\\
\bar{V}_{k+1/2} &= \eta_{t_{k+1/2}} + A_{2_{k+1/2}}(\phi_{k+1}-\phi_k)\\
&\quad + \tfrac{1}{2}\left[A_{4_{k+1}}\delta_\xi\phi_{k+1} + A_{4_k}\delta_\xi\phi_k\right]\\
&\quad + \tfrac{1}{2}\left[A_{6_{k+1}}\delta_\zeta\phi_{k+1} + A_{6_k}\delta_\zeta\phi_k\right]\\
\bar{W}_{\ell+1/2} &= \zeta_{t_{\ell+1/2}} + A_{3_{\ell+1/2}}(\phi_{\ell+1}-\phi_\ell)\\
&\quad + \tfrac{1}{2}\left[A_{5_{\ell+1}}\delta_\xi\phi_{\ell+1} + A_{5_\ell}\delta_\xi\phi_\ell\right]\\
&\quad + \tfrac{1}{2}\left[A_{6_{\ell+1}}\delta_\eta\phi_{\ell+1} + A_{6_\ell}\delta_\eta\phi_\ell\right]
\end{aligned}\right\} \tag{18}$$

with similar treatment for $\bar{U}_{j-1/2}$, $\bar{V}_{k-1/2}$, and $\bar{W}_{\ell-1/2}$. The metric terms at these points are obtained using simple averages (e.g., $A_{1_{j+1/2}} = (A_{1_j} + A_{1_{j+1}})/2$). Here $\Delta\xi = \Delta\eta = \Delta\zeta = 1$ and only the varying indices are indicated. The derivative terms in the contravariant velocities expressions are replaced with the second-order-accurate central difference approximations

$$\left.\begin{aligned}
\phi_\xi\big|_j &\doteq \delta_\xi\phi_j = \tfrac{1}{2}(\phi_{j+1}-\phi_{j-1})\\
\phi_\eta\big|_k &\doteq \delta_\eta\phi_k = \tfrac{1}{2}(\phi_{k+1}-\phi_{k-1})\\
\phi_\zeta\big|_\ell &\doteq \delta_\zeta\phi_\ell = \tfrac{1}{2}(\phi_{\ell+1}-\phi_{\ell-1})
\end{aligned}\right\} \tag{19}$$

Stability is maintained in supersonic regions by introducing an artificial viscosity term through the use of an upwind bias of the density

$$\tilde{\rho}_{j+1/2} = (1-\nu_{j+1/2})\left(\frac{\rho_{j+1}+\rho_j}{2}\right) + \nu_{j+1/2}\left[\frac{(1+\theta)}{2}\rho_{j+r} + \frac{(1-\theta)}{2}\rho_{j-1+3r}\right] \tag{20}$$

where $r = 0$ or 1 for $U < $ or > 0. The parameter $\theta = 2$ for second-order spatial accuracy in supersonic regions, and $\theta = 0$ for first-order accuracy. The switching parameter ν is defined by

$$\nu_{j+1/2} = \begin{cases} \max[1-(\rho/\rho^*)^2_j, 0]C & U_{j+1/2} > 0 \\ \max[1-(\rho/\rho^*)^2_{j+1}, 0]C & U_{j+1/2} < 0 \end{cases}$$

with $1 \le C \le 10$. Note that upwinding is used in the ξ direction only at present. For the computations performed to date this has been satisfactory, but in general it will be necessary to add it in the other directions as well (see Ref. 17 for the extension).

The metric quantities on the right-hand side of Eq. (17) are computed using three-point, second-order-accurate difference expressions

$$\left.\begin{aligned}
x_\xi &= \tfrac{1}{2}(x_{j+1}-x_{j-1})\\
x_\eta &= \tfrac{1}{2}(x_{k+1}-x_{k-1})\\
x_\zeta &= \tfrac{1}{2}(x_{\ell+1}-x_{\ell-1})
\end{aligned}\right\} \tag{21}$$

with similar treatment for the y and z metric terms. At the boundaries, 3-point, one-sided difference expressions are used.

It is necessary to subtract a numerical truncation error term because of an incomplete metric cancellation.[13,18,19] If the velocity and density are set to free-stream conditions then the terms in Eq. (17) will not be identically zero, but will be proportional to the numerical truncation error in the differencing of the metrics. This is caused by the choice of differencing used for the metric quantities in Eq. (21) and the spatial derivative terms in Eq. (17). The error term can be quite appreciable

for a highly stretched grid. Let R_∞ represent the truncation error term obtained by setting $\phi = M_\infty x$ and $\rho = 1$ initially and computing the terms in Eq. (17), i.e.,

$$R_\infty = \bar{\delta}_\xi\left(\frac{\bar{U}_\infty}{J}\right) + \bar{\delta}_\eta\left(\frac{\bar{V}_\infty}{J}\right) + \bar{\delta}_\zeta\left(\frac{\bar{W}_\infty}{J}\right) \tag{22}$$

This error term is then subtracted from the right-hand side of Eq. (15) at each time step. This correction is particularly effective in the highly stretched, coarse-grid far-field where the solution is close to free stream. Near the airfoil the grid resolution should be sufficiently fine that R_∞ is negligible.

Approximate Factorization

To avoid costly matrix inversions at each time level Eq. (15) is approximately factored into L_ξ, L_η, and L_ζ operators

$$\left[I + hU^n\delta_\xi - \frac{h^2}{\hat{\beta}^n}\bar{\delta}_\xi(\hat{\rho}A_1)^n\bar{\delta}_\xi\right]$$
$$\times\left[I + hV^n\delta_\eta - \frac{h^2}{\hat{\beta}^n}\bar{\delta}_\eta(\hat{\rho}A_2)^n\bar{\delta}_\eta\right]$$
$$\times\left[I + hW^n\delta_\zeta - \frac{h^2}{\hat{\beta}^n}\bar{\delta}_\zeta(\hat{\rho}A_3)^n\bar{\delta}_\zeta\right](\phi^{n+1} - \phi^n)$$
$$= (\phi^n - \phi^{n-1}) + \frac{\hat{\beta}^{n-1}}{\hat{\beta}^n}(\phi^n - 2\phi^{n-1} + \phi^{n-2})$$
$$+ \frac{h}{\hat{\beta}^n}(\hat{\rho}^n - \hat{\rho}^{n-1}) + h\frac{\hat{\beta}^{n-1}}{\hat{\beta}^n}(U^{n-1}\delta_E + V^{n-1}\delta_\eta$$
$$+ W^{n-1}\delta_\zeta)(\phi^n - \phi^{n-1}) + \frac{h^2}{\hat{\beta}^n}\left[\bar{\delta}_\xi\left(\frac{\tilde{\rho}\bar{U}}{J}\right)^n + \bar{\delta}_\eta(\hat{\rho}\bar{V})^n\right.$$
$$\left. + \bar{\delta}_\zeta(\hat{\rho}\bar{W})^n - R_\infty\right] \tag{23}$$

This equation has the form

$$L_\xi L_\eta L_\zeta(\phi^{n+1} - \phi^n) = R \tag{24}$$

and it is implemented as algorithm as

$$\left.\begin{aligned} L_\xi\Delta\phi^* &= R \\ L_\eta\Delta\phi^{**} &= \Delta\phi^* \\ L_\zeta\Delta\phi^{n+1} &= \Delta\phi^{**} \\ \phi^{n+1} &= \phi^n + \Delta\phi^n \end{aligned}\right\} \tag{25}$$

The algorithm Eq. (25) requires only a series of scalar, tridiagonal inversions and it is therefore very efficiently solved. Computer storage for three levels of ϕ and one level of ρ are required.

Steady State Algorithm

A fast, steady state ADI relaxation algorithm is readily obtained from Eq. (23) by turning off the unsteady terms on the right-hand side and the space-time derivative terms on the left-hand side with a simple flag giving

$$[I - h\bar{\delta}_\xi(\hat{\rho}A_1)^n\bar{\delta}_\xi][I - h\bar{\delta}_\eta(\hat{\rho}A_2)^n\bar{\delta}_\eta][I - h\bar{\delta}_\zeta(\hat{\rho}A_3)^n\bar{\delta}_\zeta]$$
$$\times(\phi^{n+1} - \phi^n) = h\left[\bar{\delta}_\xi\left(\frac{\tilde{\rho}\bar{U}}{J}\right)^n + \bar{\delta}_\eta(\hat{\rho}\bar{V})^n + \bar{\delta}_\zeta(\hat{\rho}\bar{W})^n - R_\infty\right] \tag{26}$$

This is satisfactory for subcritical and slightly critical cases, but additional temporal damping will be required for cases involving large regions of supersonic flow. For these cases, the space-time derivatives on the left-hand side of Eq. (23) may be turned back on with proper upwind bias to provide the necessary temporal damping and yield steady state performance that approaches the AF2 scheme.

Boundary Condition Implementation

The body-surface flow tangency condition is imposed by using Eq. (8) to derive a space-time extrapolation procedure for updating the solution on the body. A second-order accurate, finite-difference approximation to Eq. (8) can be derived by replacing the ξ and η spatial derivative terms with central difference operators and the ζ derivative with a three-point one-sided difference expression. The resulting equation is written in delta form and approximately factored to yield

$$\left(I - \frac{2}{3A_3}\delta_\xi\right)\left(I - \frac{2}{3A_3}\right)\delta_\eta\,\Delta\phi_1^{n+1}$$
$$= \frac{1}{3}(\phi_2^{n+1} - \phi_3^{n+1}) + \frac{2}{3A_3}[\zeta_t + (A_5\delta_\xi + A_6\delta_\eta)\phi_2^{n+1}] \tag{27}$$

where $\Delta\phi_1^{n+1} = \phi_{\ell=1}^{n+1} - \phi_{\ell=2}^{n+1}$. At the trailing edge, two-point one-sided difference operators are used for δ_ξ. Similarly, at the symmetry plane a one-sided operator is used for δ_η. After the interior flow field has been updated, Eq. (27) is then solved to update the solution on the body.

The governing equation is solved on the symmetry plane for wing cases using the warped, cylindrical coordinate system shown in Fig. 1. Along this boundary flow tangency is imposed by setting the fluxes on each side of the wall (k = 1) to be equal and opposite, viz

$$\left(\frac{\rho V}{J}\right)_{k=1/2} = -\left(\frac{\rho V}{J}\right)_{k=3/2}$$

For this grid topology, the solution for the velocity on the wing extension (i.e., the flat-plate section beyond the wing tip) is determined by averaging the solution from known updated values one surface away ($\ell = 2$). In particular, the potential is set to be the free-stream value on the flat-plate section (i.e., $\phi_\infty(x,0,z)$) plus the average of the perturbations from free-stream values at the points immediately above and below. The density is also extrapolated and averaged on the wing extension.

The new values of the circulation Γ in the wake are obtained by solving Eq. (9) after the velocity potential has been updated in the flow field and on the body surface. Equation (9) is

solved using a finite-difference approximation where ∂_η and ∂_ζ are replaced by the central and backward difference operators, δ_η and $\breve{\delta}_\zeta$, respectively. The equation is written in delta form as

$$(I + \Delta t(W)^n + \Delta t(V)^n\delta_\eta)\Delta\Gamma^n$$

$$= -\Delta t((V)^n\delta_\eta + (W)^n\breve{\delta}_\zeta)\Gamma^n + \Delta t(W)^n\Delta\Gamma^n_{\ell-1} \qquad (28)$$

Equation (28) is used to update Γ to the new time level. The trailing-edge value of Γ is obtained by setting it equal to the new value of the jump in potential there. The solution to Eq. (28) is then computed by marching in the ζ direction, thus solving tridiagonals in the η direction at each ζ = constant line.

For nonlifting flows, the outer boundary remains fixed at free-stream conditions. For steady lifting flows, the values of the potential are updated using the compressible vortex solution

$$\phi = M_\infty x + \frac{\Gamma}{2\pi}\theta \qquad (29)$$

In the preliminary calculations obtained, the outer boundary values for the potential in the unsteady case remain unchanged from their steady-state values. Care is taken to place the boundary sufficiently far away to prevent reflected waves from contaminating the solution at the body.

Density Update

After the velocity potential has been updated in the entire flow field using Eq. (25) and on the boundaries and wake using Eqs. (27), (28), and (29), the density is updated. The density is computed from the Bernoulli relation, Eq. (9), where the spatial derivatives are replaced with the difference expressions of Eq. (19) and the time derivative with a two-point backward difference operator. At the body surface, an expression for the ζ derivative is determined by solving Eq. (9) for ϕ_ζ, i.e.,

$$\phi_\zeta = -\frac{1}{A_3}(\zeta_t + A_5\phi_\xi + A_6\phi_\eta)$$

and is also used in the computation of the contravariant velocities on the body. At the wall boundary (y = 0), a two-point, one-sided difference expression was used for the η derivative. All exponential functions were eliminated by using binomial expansions.

Grid Generation

A number of grid-generation programs have been used in the present investigation because of the wide range of geometries considered. The O-type grids used for the airfoil calculations were obtained using the grid-generation program GRAPE (grids about airfoils using Poisson's equation).[20] This grid-generation scheme numerically generates solutions to Poisson's equation to establish regular and smooth finite-difference meshes around arbitrary two-dimensional bodies. The inhomogeneous terms are automatically chosen to control the mesh point spacing adjacent to the boundaries and the angles with which mesh lines intersect the boundaries. The equations are transformed to and solved in the computational domain using successive overrelaxation (SLOR).

The grid generation program used for wing cases that use the warped cylindrical coordinates was presented in Ref. 17. The finite-difference mesh is generated by employing a standard two-dimensional grid-generation scheme similar to GRAPE. Numerical solutions to the elliptic, partial differential grid-generation equations are obtained in each spanwise plane used as a defining station for the wing. The equations are transformed to and solved in the computational domain using a fast approximate-factorization algorithm. This establishes values for x and z in each spanwise plane. The coordinate values in the spanwise direction (y values) are computed from a stretching formula that in its simplest form gives equal spacing over the wing with relatively rapid stretching beyond the tip. For the wing extension, a flat-plate section is used.

For the three-dimensional flows using warped spherical coordinates, a hyperbolic grid-generation procedure was used (Steger, J. L., Jespersen, D., and Strigberger, J., private communication). This procedure is a three-dimensional extension of the scheme devised by Steger and Chaussee[21] in which a system of hyperbolic equations is solved using an implicit, marching finite-difference scheme. Two of the equations are derived from orthogonality conditions and the third relation is a specification of the mesh cell volume. Since this is a noniterative algorithm, very fast grid generation is obtained.

IV. Results

The algorithm Eq. (25) has been coded into a computer program named TUNA (transonic unsteady aerodynamics). The three-dimensional computer code functions as a two-dimensional code by simply setting a flag. Consequently, steady-state calculations in two dimensions were first performed to verify the accuracy of the steady and unsteady algorithms. The solutions obtained with the present method, for several standard test cases, have been compared with the 2D steady, transonic computer code TAIR (transonic airfoil analysis)[22,23] which solves the steady full-potential equation using the approximate factorization scheme AF2.

A comparison of solutions for a subcritical nonlifting test case is shown in Fig. 2. The pressure distribution on the upper surface of an NACA 0012 airfoil at $M_\infty = 0.72$ and $\alpha = 0°$ shows excellent agreement between the two codes. Both codes used 100×31 O-mesh type grids with TAIR being internally generated. The grid used in the present method was generated using GRAPE (Fig. 3). This steady-state computation was performed using the steady and unsteady algorithms. An optimum set of acceleration parameters was found for the steady algorithm and an optimum time step for the unsteady algorithm. The residual histories for these computations are shown in Fig. 4. The new steady scheme displays rapid steady-state convergence and is at least an order of magnitude faster than the unsteady scheme. The present method dropped the maximum residual four orders of magnitude in 43 iterations compared to 74 for TAIR. Past experience has shown that the ADI convergence rate is about twice as fast as AF2 for subcritical flows.

The subcritical lifting results (Fig. 5) for a NACA 0012 airfoil at $M_\infty = 0.63$ and $\alpha = 2°$ indicate that the circulation model is suitable. The same grids described in the previous case are again used here. There is a slight discrepancy between the two codes in the amount of leading-edge suction. This is most likely a result of differences in the grid, in particular, the amount of grid-point clustering around the leading edge. The lift coefficients for TUNA and TAIR were 0.338 and 0.334, respectively.

As a final two-dimensional steady-test case, supercritical lifting-flow solutions are compared in Fig. 6 for a NACA 0012 at $M_\infty = 0.75$ and $\alpha = 2°$. A finer grid with dimensions of 151×31 was used for the TAIR computation. The two solutions are in good agreement and the shock profile for the present method is quite sharp despite the coarser grid. The lift coefficients for TUNA and TAIR for these two computations were 0.5963 and 0.5881, respectively. The TAIR solution for each of the airfoil cases does not compute a stagnation point at the trailing edge, as in the present method, because the density is extrapolated to avoid potential problems from the grid singularity. The present method does not extrapolate the density for these computations but it has been found that in some fine-grid cases this causes oscillations and overshoots near the trailing edge.

It is worth noting that for supercritical cases, the AF2 algorithm has been shown to provide faster convergence rates than ADI.[24] But as previously discussed, the judicious addition of certain temporal damping terms may provide convergence rates for ADI that approach AF2. These modifications, as yet, have not been tested. Convergence-rate comparisons for supercritical flows with large regions of supersonic flow and strong shock waves have indicated that AF2 converges 2 to 3 times faster than the present ADI scheme.

Unsteady flow results are shown in Fig. 7 for an NACA 64A010 airfoil sinusoidally oscillating in plunge +l about $\alpha = 0°$. The reduced frequency, $k = \omega c/U_\infty$, is 0.4 and $M_\infty = 0.80$. The lift coefficient is plotted vs time during the fourth cycle of oscillation. Also shown are computed results from an Euler equation solution.[18] The magnitudes of the lift are in excellent agreement, with only about a 1% discrepancy. In addition, the computed phases are in good agreement. The Euler solution displays a phase lag of approximately 35° in comparison to 38° as predicted by TUNA. Also shown are pressure coefficient distributions at three times corresponding to zero lift and the maximum (positive and negative) values of the lift.

Three-dimensional, steady-state computations were performed for a rectangular wing of aspect ratio 6 with an NACA 0012 airfoil section and compared with the 3D transonic computer code TWING (transonic wing analysis) that solves the steady, full-potential equation using the AF2 scheme. The identical warped cylindrical grid system sketched in Fig. 1 was used for the computations of the computer codes TUNA and TWING. The dimensions were 100×10×30 with 5 span stations on the wing, which is a reasonably coarse grid in the span direction.

A comparison of solutions for a subcritical lifting case with $M_\infty = 0.63$ and the wing at $\alpha = 2°$ is shown in Fig. 8. The pressure distribution on the wing for all five span stations shows excellent agreement between the two codes. The lift coefficients at the wall station for TUNA and TWING were 0.250 and 0.253, respectively, which represents about a 25% reduction from the two-dimensional results previously obtained. TUNA and TWING converged the maximum residual two orders of magnitude for this case in 85 and 100 iterations, respectively.

The results for a supercritical lifting solution with $M_\infty = 0.75$ and the wing at $\alpha = 2°$ are compared in Fig. 9. The two solutions are in reasonably good agreement. At each span station, there is a slightly larger expansion computed by TUNA from the leading-edge region of the airfoil section to the shock position, which is located about 2.5% farther upstream than that computed by TWING. This discrepancy may be due to the different ways in which each code determines the solution on the wing extension. As mentioned before, TUNA extrapolates and averages the solution there while TWING solves the full-potential equation on this section. The shock profiles computed by TWING appear to be sharper and the reexpansion singularity is not captured by TUNA.

The two-dimensional grids employed here are O-type grids. Even though these grids require special attention at the trailing edge, a result of the mapping singularity, they are preferred over C- and H-type meshes because they are more grid-point efficient, especially in the far-field. Similarly, warped spherical grids offer the same advantage in grid-point efficiency in three dimensions over current wing-type grids that are typically constructed in a manner similar to warped cylindrical grids used for the previous three-dimensional calculations. The use of spherical grids requires special treatment to handle the mapping singularity on the axes. This capability has been incorporated in the present method.

To demonstrate the ability to employ spherical grids, the three-dimensional incompressible flow over a sphere is presented as a simple check case. A comparison between the exact incompressible flow solution and the present method with $M_\infty = 0.01$ is shown in Fig. 10. A 40×21×22 radially stretched spherical grid was used. The solution shown is the pressure distribution on the upper surface of the sphere from leading to trailing edge in the plane of symmetry perpendicular to the axes. The agreement is correct to plottable accuracy. The maximum residual converged six orders of magnitude in 30 iterations for this case.

A more complicated geometry using a spherical grid topology is an ellipsoid type wing with a chord of 1, aspect ratio of 4, and thickness of 1/4. Here, as with the sphere, the axes of the spherical grid are aligned with the y axis and the flow is in the x-direction. The surface grid-point distribution and other partial views of the grid are shown in Fig. 11. The pressure coefficient on the upper surface in the $y = 0$ plane is shown in Fig. 12 for $M_\infty = 0.77$. A relatively strong shock can be seen at about 85% chord. The shock profile is not particularly sharp but the grid is fairly coarse with only 60 points around the body in the ξ direction. The residual converged 3 orders of magnitude in 100 iterations for this case.

The current version of TUNA has been programmed and run on the CRAY 1S. The program requires

approximately 0.0001 sec of CPU time per grid point per iteration. The code is presently in an inefficient state and no effort has been made to take advantage of the vector capabilities of the CRAY.

V. Concluding Remarks

An implicit, approximate-factorization, finite-difference algorithm has been developed for solving the conservative full-potential equation for computing steady and unsteady transonic flows in two- and three-dimensions. A general coordinate transformation was employed for the simple and accurate treatment of arbitrary body surfaces. This algorithm has been programmed into a computer code named TUNA. Results have been presented for several two- and three-dimensional steady and two-dimensional unsteady flows to verify the accuracy of the algorithm.

Acknowledgments

The effort of Prof. Steger was supported under Air Force Flight Dynamics Contract F33615-81-K-3020.

References

[1]Farmer, M. G. and Hanson, P. W., "Comparison of Supercritical and Conventional Wing Flutter Characteristics," Proceedings of the AIAA/ASME/SAE 17th Structures, Structural Dynamics, and Materials Conference, King of Prussia, Pa., April 1976, pp. 608-611.

[2]Ashley, H., "On the Role of Shocks in the 'Sub-Transonic' Flutter Phenomena," Journal of Aircraft, Vol 17, March 1980, pp. 187-197.

[3]Ballhaus, W. F., and Steger, J. L., "Implicit Approximate-Factorization Schemes for the Low-Frequency Transonic Equation," NASA TM X-73,082, Nov. 1975.

[4]Ballhaus, W. F., and Goorjian, P. M., "Implicit Finite Difference Computations of Unsteady Transonic Flows about Airfoils," AIAA Journal, Vol. 15, Dec. 1977, pp. 1728-1735.

[5]Houwink, R., and van der Vooren, J., "Improved Version of LTRAN2 for Unsteady Transonic Flow Computations," AIAA Journal, Vol. 18, Aug. 1980, pp. 1008-1010.

[6]Hessenius, K. A., and Goorjian, P. M., "A Validation of LTRAN2 with High Frequency Extensions by Comparison with Experimental Measurements of Unsteady Transonic Flows," NASA TM-81307, July 1981.

[7]Chow, L. J., and Goorjian, P. M., "Implicit Unsteady Transonic Airfoil Calculations at Supersonic Freestreams," AIAA Paper 82-0934, presented at the AIAA/ASME Joint Thermophysics, Fluids, Plasma and Heat Transfer Conference, St. Louis, Missouri, June 7-11, 1982.

[8]Rizzetta, D., "Procedures for the Computation of Unsteady Transonic Flows Including Viscous Effects," NASA CR-166249, Jan. 1982.

[9]Borland, C., Rizzetta, D., and Yoshihara, H., "Numerical Solution of Three-Dimensional Unsteady Transonic Flow Over Swept Wings," AIAA Paper 80-1369, July 1980.

[10]Borland, C. J., and Rizzetta, D. P., "Nonlinear Transonic Flutter Analysis," AIAA Paper 81-0608-CP, presented at the AIAA Dynamic Specialist Conference, Atlanta, Ga., April 9-10, 1981.

[11]Guruswamy, P., and Goorjian, P. M., "Comparisons Between Computations and Experimental Data in Unsteady Three-Dimensional Transonic Aerodynamics, Including Aeroelastic Applications," AIAA Paper 82-0690-CP, presented at the AIAA/ASME/ASCE/AHS 23rd Structures, Structural Dynamics, and Materials Conference, May 10-12, 1982.

[12]Goorjian, P. M., "Implicit Computations of Unsteady Transonic Flow Governed by the Full Potential Equation in Conservation Form," AIAA Paper 80-0150, Jan. 1980.

[13]Steger, J. L., and Caradonna, F. X., "A Conservative Implicit Finite Difference Algorithm for the Unsteady Transonic Potential Equation," AIAA Paper 80-1368, July 1980.

[14]Chipman, R., and Jameson, A., "Alternating-Direction Implicit Algorithm for Unsteady Potential Flow," AIAA Journal, Vol. 20, Jan. 1982, pp. 18-24.

[15]Sankar, N. L., and Tassa, Y., "An Algorithm for Unsteady Transonic Potential Flow Past Airfoils," paper presented at the Seventh International Conference on Numerical Methods in Fluid Dynamics, June 1980.

[16]Sankar, N. L., Malone, J. B., and Tassa, Y., "An Implicit Conservative Algorithm for Steady and Unsteady Three-Dimensional Transonic Potential Flows," AIAA Paper 81-1016, June 1981.

[17]Holst, T. L., and Thomas, S. D., "Numerical Solution of Transonic Wing Flow Fields," AIAA Paper 82-0105, paper presented at the AIAA 20th Aerospace Sciences Meeting, Orlando, Florida, Jan. 11-14, 1982.

[18]Steger, J. L., "Implicit Finite Difference Solution of Flow About Arbitrary Two-Dimensional Geometries," AIAA Journal, Vol. 16, 1978.

[19]Pulliam, T. H., and Steger, J. L., "On Implicit Finite Difference Simulations of Three-Dimensional Flow," AIAA Paper 78-10, Jan. 1978.

[20]Sorenson, R. L., "A Computer Program to Generate Two-Dimensional Grids About Airfoils and Other Shapes by Use of the Poisson's Equation," NASA TM-81198, May 1981.

[21]Steger, J. L., and Chaussee, D. S., "Generation of Body Fitted Coordinates Using Hyperbolic Partial Differential Equations," SIAM Journal Sci. Stat. Comput., Vol. 1, No. 4, Dec. 1980, pp. 431-437.

[22]Holst, T. L., "An Implicit Algorithm for the Conservative, Transonic Full Potential Equation Using an Arbitrary Mesh," AIAA Journal, Vol. 17, Oct. 1979, pp. 1038-1045.

[23]Dougherty, F. C., Holst, T. L., Gundy, K. L., and Thomas, S. D., "TAIR – a Transonic Airfoil Analysis Computer Code," NASA TM-81296, May 1981.

[24]Holst, T. L., and Ballhaus, W. F., [illegible], Conservative Schemes for the Full Potential Equation Applied to Transonic Flows," AIAA Journal, Vol. 17, Feb. 1979, pp. 145-152.

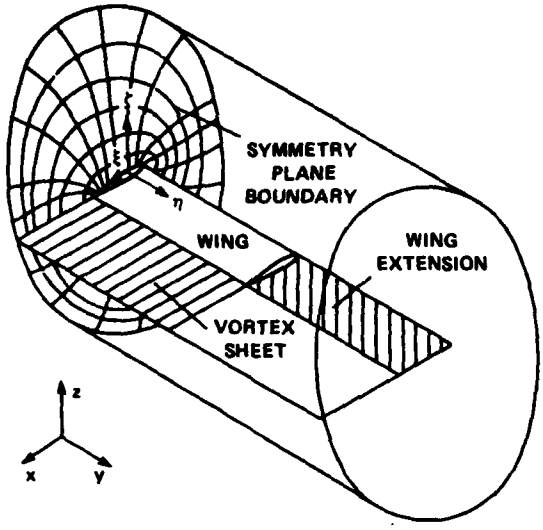


Fig. 1 Schematic of cylindrical grid system used for wing calculations employing a plane of symmetry.

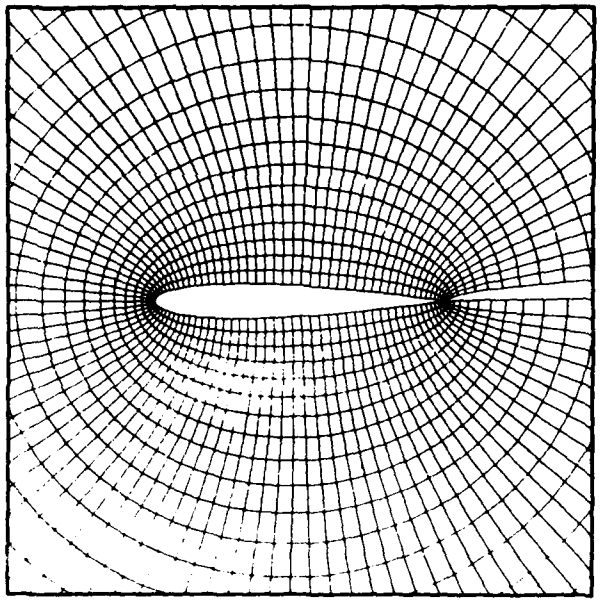


Fig. 3 O-mesh type finite difference grid for an NACA 0012 airfoil numerically generated using GRAPE code (100×31 grid points).

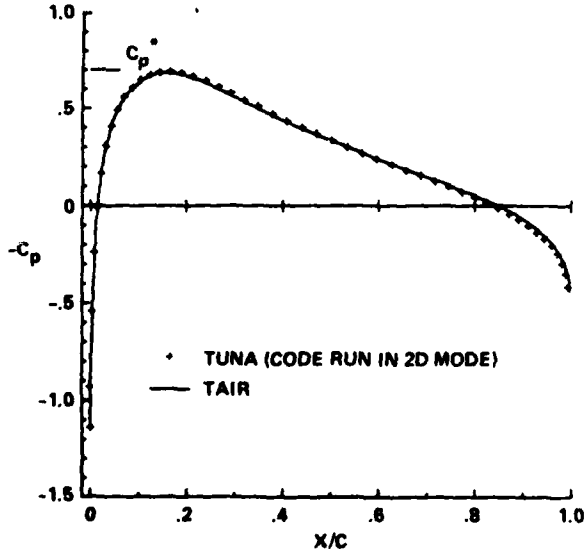


Fig. 2 Two-dimensional pressure coefficient distribution comparison: NACA 0012 airfoil, $M_\infty = 0.72$, $\alpha = 0°$.

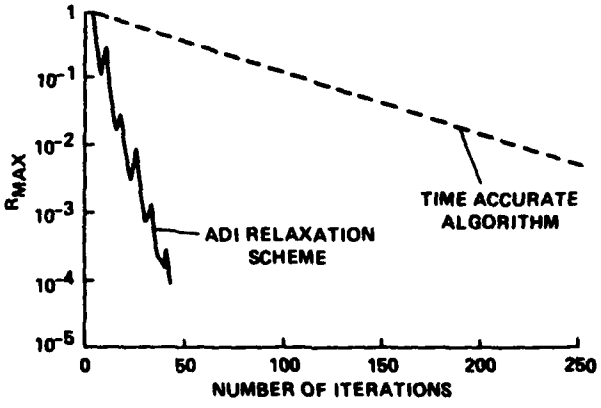


Fig. 4 Residual history comparison between the time accurate algorithm and ADI relaxation option: NACA 0012 airfoil, $M_\infty = 0.72$, $\alpha = 0°$.

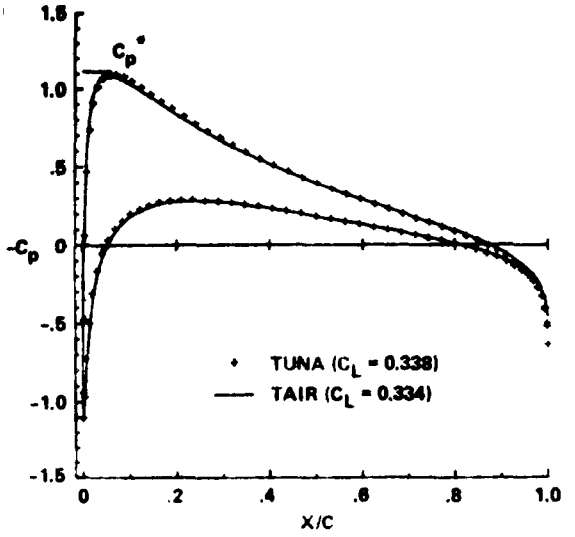


Fig. 5 Two-dimensional pressure coefficient distribution comparison: NACA 0012 airfoil, M_∞ = 0.63, α = 2°.

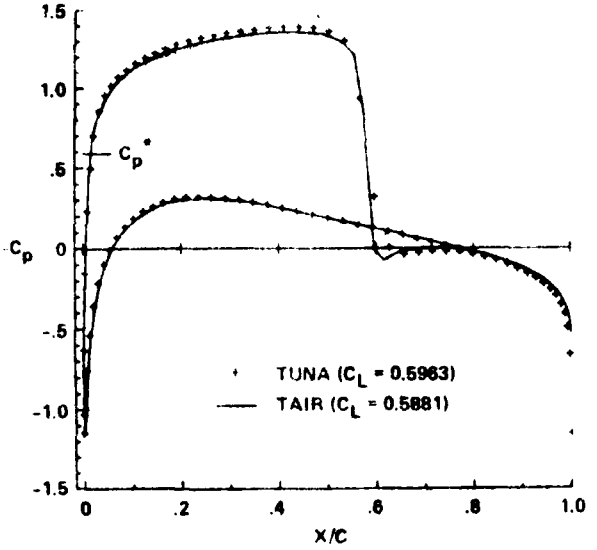


Fig. 6 Two-dimensional pressure coefficient distribution comparison: NACA 0012 airfoil, M_∞ = 0.75, α = 2°.

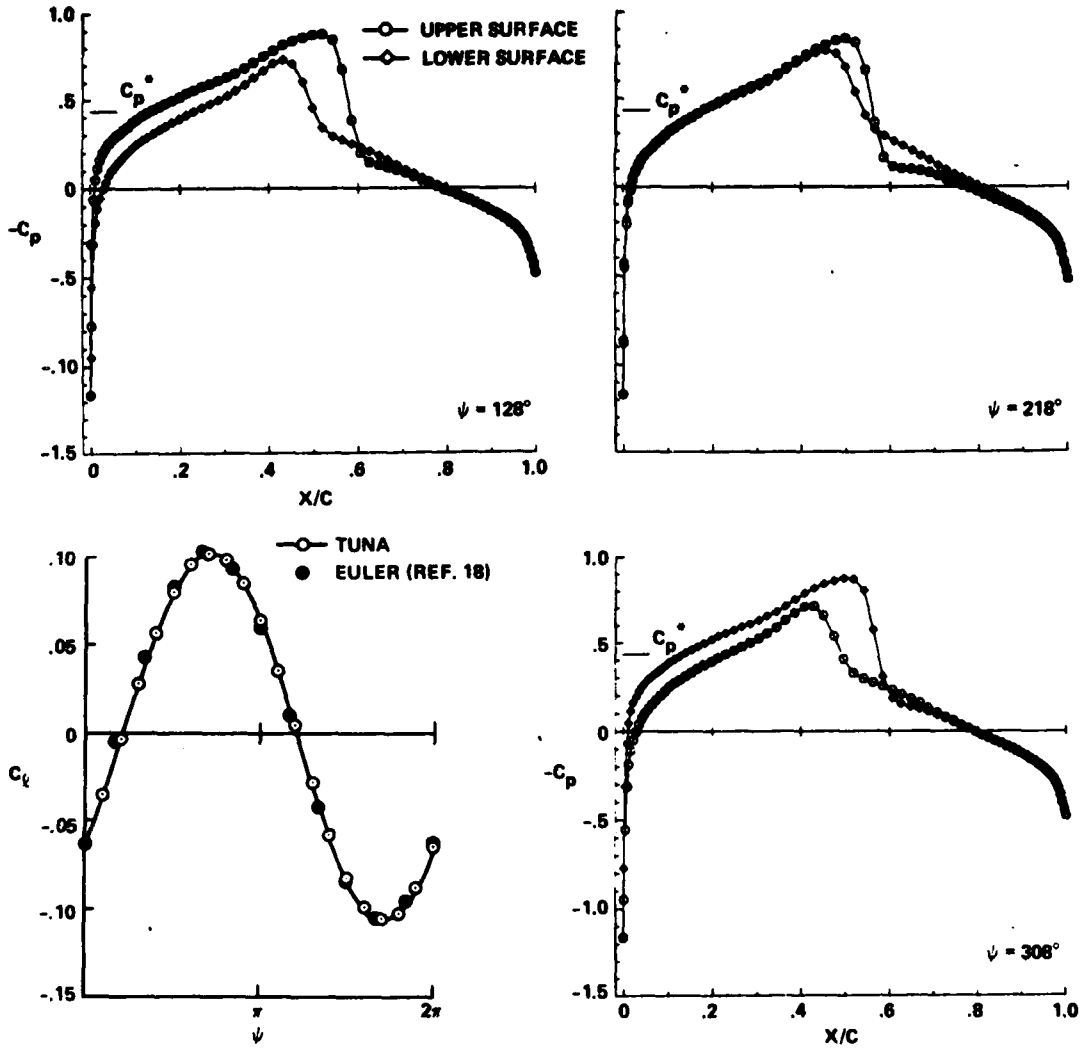


Fig. 7 Time variation of the lift coefficient and various pressure coefficient distributions for an NACA 64A010 airfoil sinusoidally oscillating in plunge, $M_\infty = 0.80$, $k = 0.40$, $\alpha_p = 1° \sin(kM_\infty t)$.

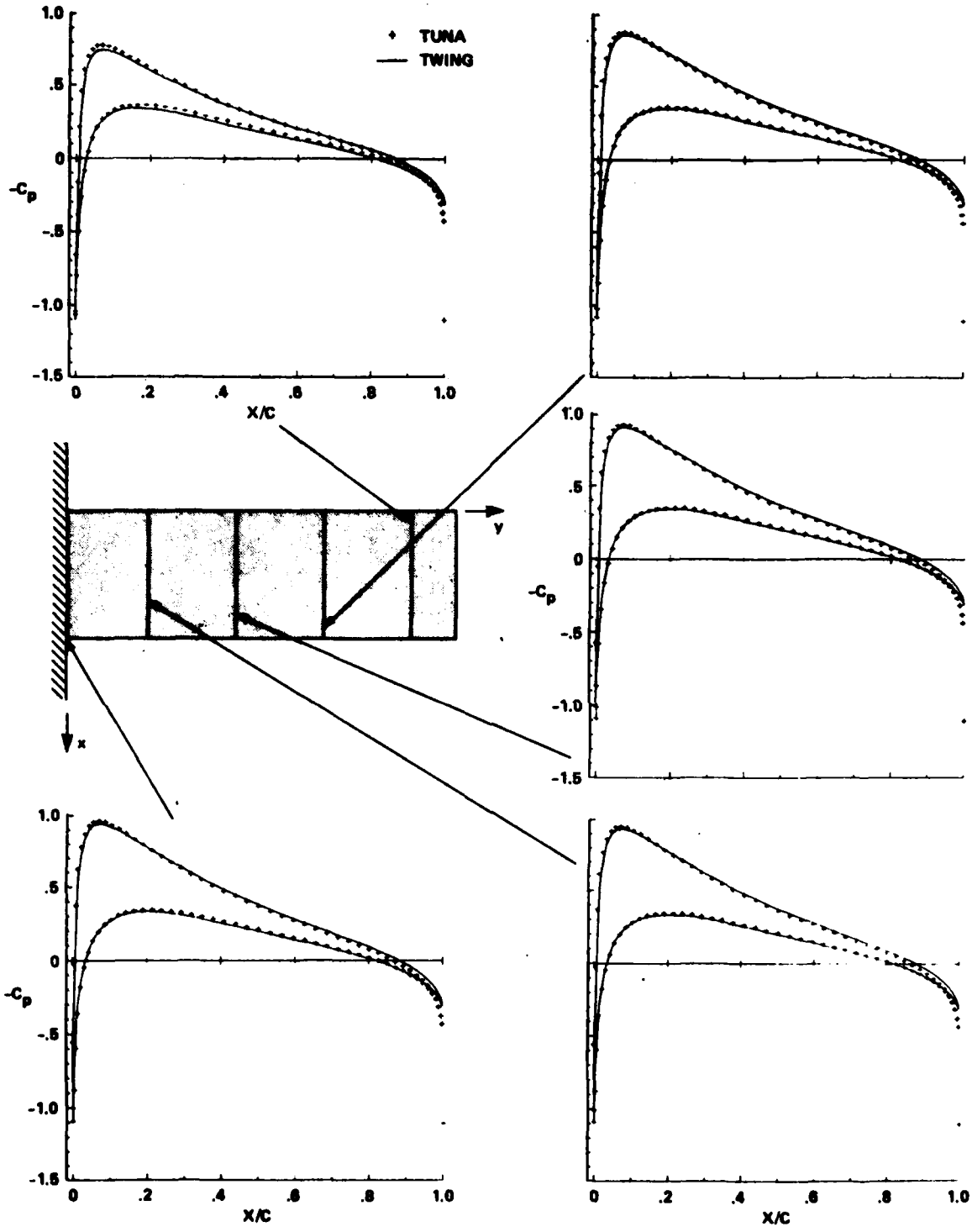


Fig. 8 Section pressure coefficient distribution comparisons for an NACA 0012 rectangular wing, $M_\infty = 0.63$, $\alpha = 2°$, $AR = 6$.

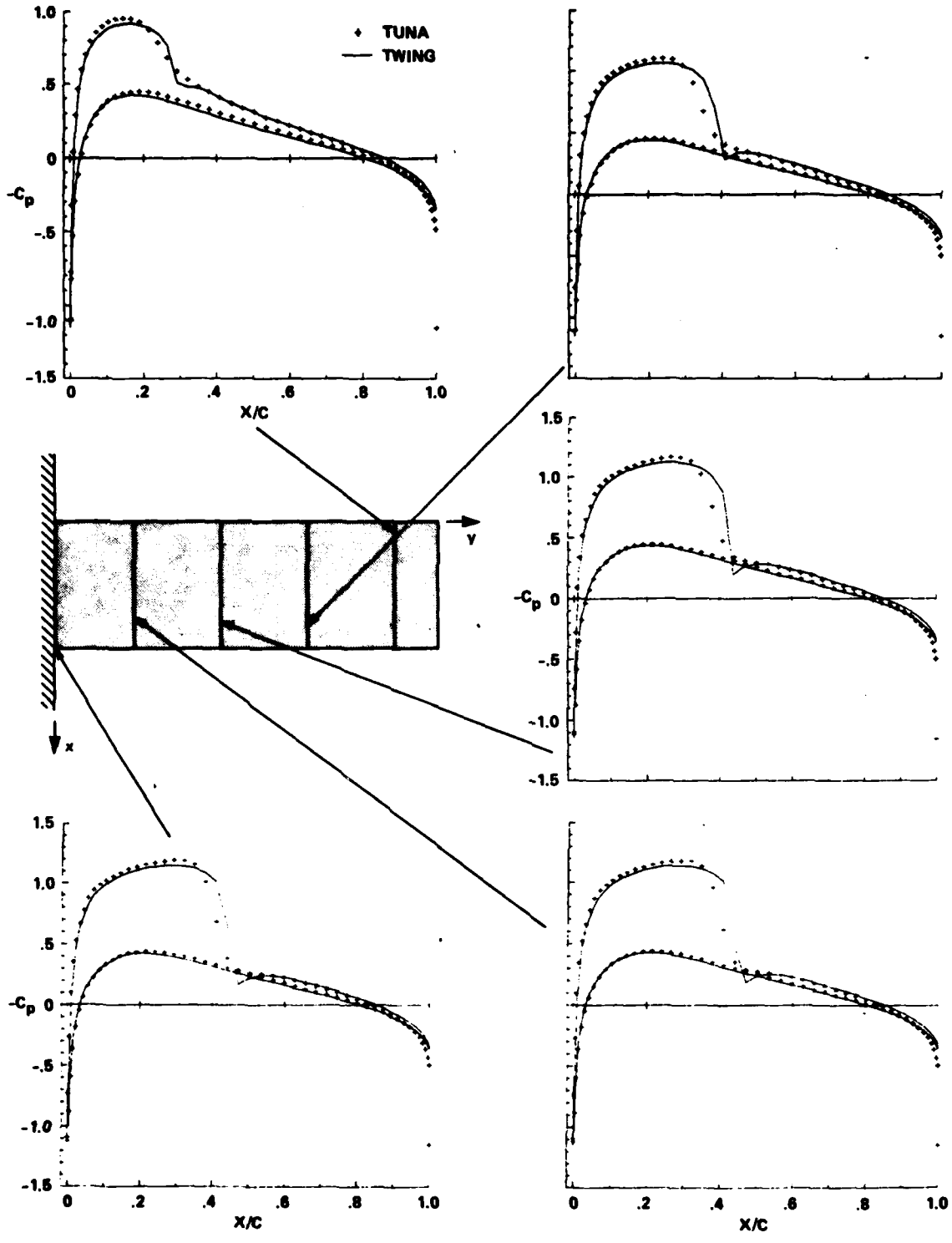


Fig. 9 Section pressure coefficient distribution comparisons for an NACA 0012 rectangular wing, $M_\infty = 0.75$, $\alpha = 2°$, $AR = 6$.

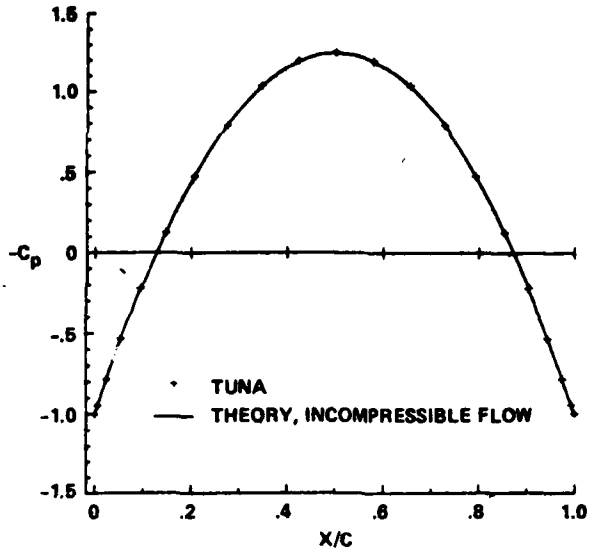


Fig. 10 Comparison of computed pressure coefficient with incompressible flow theory for flow about a about a sphere.

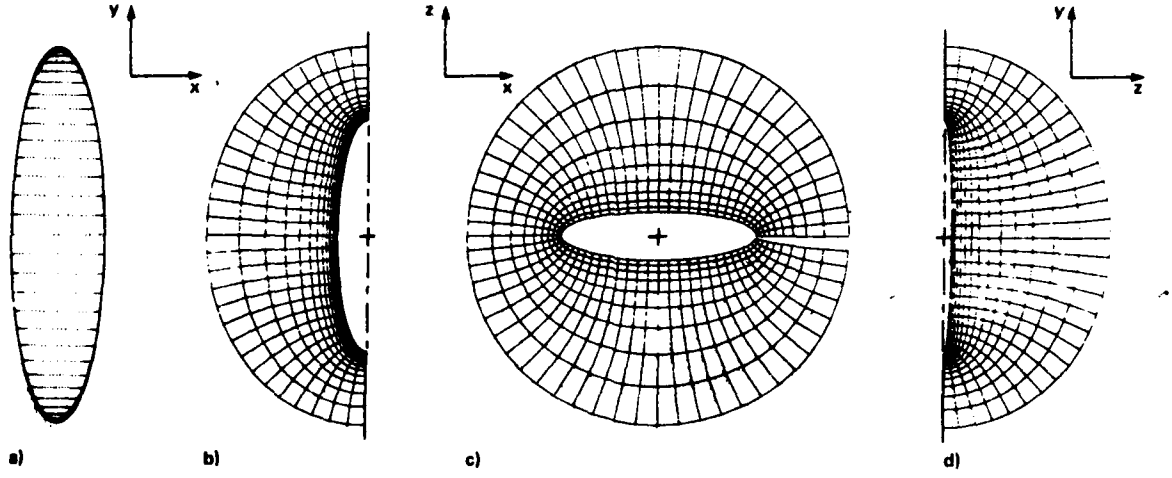


Fig. 11. Spherical grid used for ellipsoidal wing. a) Planform view of surface grid point distribution; Partial grid views of b) Symmetry plane (z = 0), c) Midspan symmetry plane (y = 0), and d) Midchord symmetry plane.

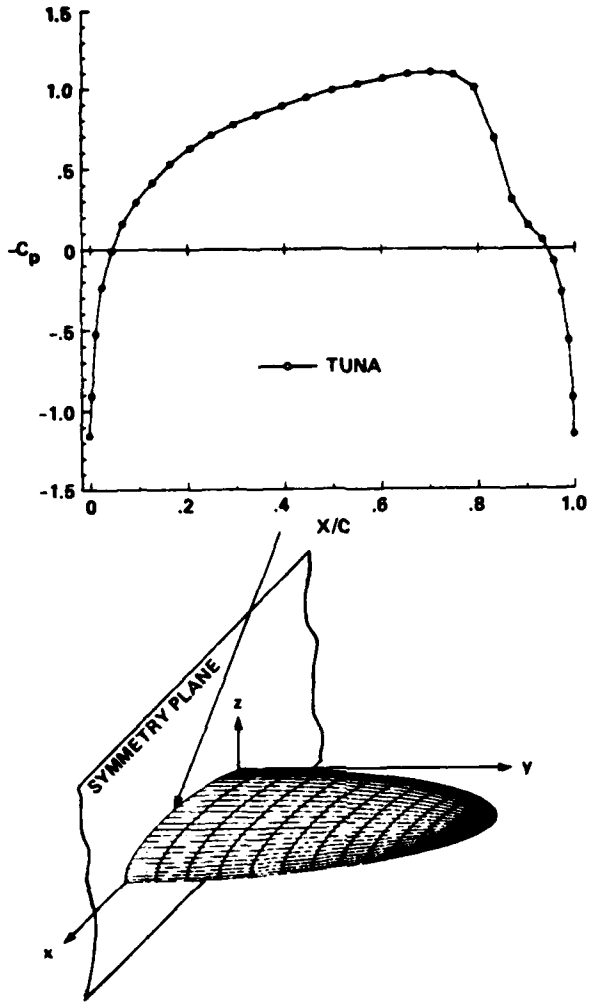


Fig. 12 Computed midspan pressure coefficient distribution for an ellipsoidal wing at $M_\infty = 0.77$, $\alpha = 0°$.

APPENDIX E

A ZONAL APPROACH FOR THE STEADY TRANSONIC SIMULATION OF INVISCID ROTATIONAL FLOW

Neal M. Chaderjian and Joseph L. Steger
Department of Aeronautics and Astronautics
Stanford University, Stanford, CA

AIAA 6th Computational
Fluid Dynamics Conference
July 13-15, 1983, Danvers, MA

A ZONAL APPROACH FOR THE STEADY TRANSONIC SIMULATION OF INVISCID ROTATIONAL FLOW

Neal M. Chaderjian† and Joseph L. Steger‡
Stanford University, Stanford, California

Abstract

A finite difference zonal method is developed to compute steady inviscid transonic flow by coupling a semi-flux split form of the Euler equations in a vorticity producing zone with a zone of scalar and vector (i.e., dual) potential equations. The dual potential equations permit vorticity convection, but not production, and are efficiently solved as an iteratively decoupled set of scalar equations. Zonal results presented for a nonlifting biconvex airfoil on a stretched Cartesian grid show substantial savings in CPU time compared to solving the semi-flux split Euler equations alone. The dual potential equations also provide an alternate way of treating potential flows with circulation. This has been demonstrated by computing a subcritical flow over a lifting airfoil using generalized curvilinear coordinates.

I. Introduction

The development of efficient numerical solution techniques for simulating rotational compressible flow has always been subject to two competing philosophies. In one approach the Navier-Stokes or Euler equations are programmed throughout the entire domain. The overall computer code is quite general in its applicability and the programming is straight forward in the sense that only one equation set is coded. In the other approach the flow field is partitioned into zones which use the simplest and most efficiently solved equations possible. For example the Navier-Stokes equations may be used in one zone, the nonlinear potential equations in another, and perhaps linear theory in the remaining field. This zonal approach has the potential for saving substantial amounts of the computer resource. However, a zonal code is significantly more complex to program because several computer codes must essentially be written (one for each governing equation set) and each zone must be interfaced. Moreover, considerable care must be taken in interfacing zones because poor interfacing can result in computational inefficiency thus offsetting the advantages of a zonal approach.

Whether the use of a zonal method will ultimately prove better than just using a single general equation set is a question that will likely remain moot, but several developments have occurred which in our view make the zonal approach more attractive than previously. The first development is that finite difference inviscid and viscous boundary layer interaction algorithms are becoming available for solving separated viscous flow and these schemes appear to be much faster than Navier-Stokes algorithms. The second development is the appearance of efficient methods for including rotationality effects into essentially potential-like governing equation sets. It is this second development which is of interest to us in this paper.

A zonal algorithm becomes more advantageous if potential-like equations can be generalized to simulate rotational, nonisentropic flow without loss of computational efficiency. This is because more of the flow field can be treated with the more efficient generalized potential code and fewer zones and thus fewer interface boundaries have to be introduced.

In this paper we study a two zone method for solving inviscid transonic flow without requiring the flow to be irrotational. Shock waves are captured in the first zone using a semi-flux split implicit finite difference method to solve the Euler equations in strong-conservation-law form. In this algorithm flux splitting is used in the direction along the body, and central differencing is used in the direction away from the body. Because of this differencing structure, this algorithm is readily extended to incorporate thin layer viscous terms although we do not exercise this option here. A dual potential formulation is solved in a zone away from the shock and is able to correctly convect but not generate vorticity. This second zone slightly overlaps the shock zone and extends over the remainder of the flow field. The dual potential scheme, which is similar to a formulation due to Hafez and Lovell[1], decomposes a velocity field into gradients of scalar potential and vector potential. The vector poten-

---

† Research Assistant, member AIAA.

‡ Associate Professor, member AIAA.

This paper is declared a work of the U.S. Government and therefore is in the public domain.

tial function accounts for vorticity, and as a consequence, does not require a circulation cut for a lifting airfoil as does a scalar potential formulation. The crucial advantage of the dual potential formulation over a primitive variable formulation for the Euler equations is that the resulting dual potential equations are weakly coupled. They can thus be treated in scalar mode and can take advantage of all of the numerical efficiencies developed for the transonic potential equation.

In the following sections we discuss the governing equation sets and zonal coupling concepts. A set of boundary conditions are developed which permit vorticity convection, and a new way of treating circulation by taking advantage of the stream function-like properties of the dual potential equations are presented. Finally, the numerical algorithms are developed followed by a discussion of results and concluding remarks.

II. Zonal Formulation

Governing Equations

The flow field has been zoned between two equation sets. In regions of expected vorticity generation, here shock waves, the Euler equations are solved in strong-conservation-law form. In transformed coordinates these equations are given by

$$\partial_\xi \hat{F} + \partial_\eta \hat{G} = 0 \tag{1}$$

where

$$\hat{F} = (\xi_x F + \xi_y G)/J$$
$$\hat{G} = (\eta_x F + \eta_y G)/J$$
$$J = \xi_x \eta_y - \xi_y \eta_x$$

and

$$F = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ u(e+p) \end{pmatrix}, \quad G = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ v(e+p) \end{pmatrix}$$

In the above equations ρ is the density, u and v the Cartesian velocities in the x and y-directions, $p = (\gamma - 1)|e - \frac{1}{2}\rho q^2|$ is the pressure and e the total energy per unit volume. The fluid speed is q. The Cartesian flux vectors F and G correspond to the x and y-directions, and J is the determinant of the transformation Jacobian.

The variables have been nondimensionalized by the free stream density and velocity as

$$\tilde{\rho} = \rho/\rho_\infty$$
$$\tilde{u} = u/u_\infty$$
$$\tilde{v} = v/u_\infty$$
$$\tilde{p} = p/(\rho_\infty u_\infty^2)$$
$$\tilde{e} = e/(\rho_\infty u_\infty^2)$$

where the $\sim$ has been suppressed for convenience.

In the present test program we use small disturbance thin airfoil boundary conditions so only simple stretching transforms are needed to cluster grid points. The stretchings are of the form

$$\xi = \xi(x)$$
$$\eta = \eta(y) \tag{2}$$

Consequently,

$$\xi_y = 0 \quad , \quad \eta_x = 0$$

and Eq. (1) is very much like its Cartesian counterpart.

In the remaining part of the flow field a restricted form of the steady Euler equations are used. In nondimensional variables these are given by

continuity

$$\partial_x(\rho u) + \partial_y(\rho v) = 0 \tag{3}$$

Crocco (vorticity) equation

$$\partial_x v - \partial_y u = -(\gamma M^2)^{-1}(v\partial_x s - u\partial_y s) \tag{4}$$

constant entropy along a streamline

$$u\partial_x s + v\partial_y s = 0 \tag{5}$$

Bernoulli equation

$$\rho = \left[1 + \frac{\gamma - 1}{2} M_\infty^2 (1 - u^2 - v^2)\right]^{\frac{1}{\gamma - 1}} e^{-s} \tag{6}$$

The nondimensionalized variables are consistent with those used for the conservation-law-form equations

with nondimensional entropy $\tilde{s} = (s - s_\infty)/R$. Once again the $\sim$ has been suppressed with R the perfect gas constant, γ the ratio of specific heats, and M the Mach number.

In regions of entropy production, for example across a shock wave or in viscous layers, Eq. (5) is invalid. The simplified equations as given also assume uniform stagnation enthalpy, although it is relatively easy to incorporate this effect into the equations and to solve as well the equation

$$u\partial_x h_{st} + v\partial_y h_{st} = 0 \tag{7}$$

In order to take advantage of efficient potential flow solvers, a dual potential representation of velocity is introduced into the simplified equation set. Specifically, the velocity components are decomposed into scalar potential and vector potential functions which for two dimensions are given by

$$\begin{aligned} u &= \partial_x\phi + \partial_y\psi = \phi_x + \psi_y \\ v &= \partial_y\phi - \partial_x\psi = \phi_y - \psi_x \end{aligned} \tag{8}$$

where ψ is the third vector potential component. Introducing the dual potential velocity relations into Eqs. (3) and (4) we obtain

continuity

$$\partial_x[\rho(\phi_x + \psi_y)] + \partial_y[\rho(\phi_y - \psi_x)] = 0 \tag{9}$$

Crocco or vorticity

$$\psi_{xx} + \psi_{yy} = (\gamma M^2)^{-1}(vs_x - us_y) = -\Omega \tag{10}$$

And finally, with transformation into general coordinates

$$\begin{aligned} &\partial_\xi(\rho a_1\phi_\xi + \rho a_2\phi_\eta) + \partial_\eta(\rho a_2\phi_\xi + \rho a_3\phi_\eta) \\ &\quad = (\rho\psi_\xi)_\eta - (\rho\psi_\eta)_\xi \end{aligned} \tag{11}$$

where

$$a_1 = \frac{\xi_x^2 + \xi_y^2}{J}, \quad a_2 = \frac{\xi_x\eta_x + \xi_y\eta_y}{J}, \quad a_3 = \frac{\eta_x^2 + \eta_y^2}{J}$$

and

$$\begin{aligned} &\partial_\xi(\alpha_1\psi_\xi + \alpha_2\psi_\eta) + \partial_\eta(\alpha_2\psi_\xi + \alpha_3\psi_\eta) \\ &\quad = (\gamma M^2 J)^{-1}[v(\xi_x s_\xi + \eta_x s_\eta) - u(\xi_y s_\xi + \eta_y s_\eta)] \end{aligned} \tag{12}$$

where

$$\begin{aligned} u &= \xi_x\phi_\xi + \eta_x\phi_\eta + \xi_y\psi_\xi + \eta_y\psi_\eta \\ v &= \xi_y\phi_\xi + \eta_y\phi_\eta - \xi_x\psi_\xi - \eta_x\psi_\eta \end{aligned}$$

For the current zonal algorithm applications we will again restrict the transforms to the simple stretchings $\xi = \xi(x)$ and $\eta = \eta(y)$. However, the fully transformed dual potential equations alone will be used to solve a subcritical lifting airfoil flow in order to illustrate their applicability to flows with circulation without the need to impose special cuts in the field.

Zone Partitioning

As sketched in Fig. 1a and 1b, the conservation-law-form equations of mass, momentum, and energy are to be solved in shock regions where vorticity is generated. Ideally this conservation-law zone would be kept as small as possible as shown in Fig. 1a. This, however, will require shock wave tracking logic and consequently we are currently defining a much more generous stationary zone for Eq. (1) as sketched in Fig. 1b. In the remaining flow field the simplified governing equations are solved in terms of the dual potential variables. For uniform incoming flow entropy correction terms are only needed behind the shock wave.

Boundary Conditions

The inviscid-conservation-law equations have been restricted to a zone about the expected shock wave which overlaps a larger zone governed by the dual potential equations (see Fig. 1b). Except along the body surface, all boundary conditions for the conservation-law zone can be supplied from the overlapping dual potential zone. Along the body surface we impose the tangency condition

$$v = u\left(\frac{dy}{dx}\right)_{airfoil} \tag{13}$$

and use linear extrapolation to supply the three remaining variables. A thin airfoil approximation is used by

imposing these boundary conditions at $y = 0$ rather than on the body surface. This boundary treatment permits verification of the method without unduely complicating the computer code. This will also lead to some solution discrepancies when results are compared to more exact theories.

The dual potential equations resolve the flow in the remaining domain. Representing the velocity by derivatives of scalar functions allows some freedom in the choice of boundary conditions. We will therefore describe the dual potential boundary conditions for our specific application which is the transonic flow over a symmetric airfoil at zero angle attack (see Fig. 2).

In prescribing a set of boundary conditions for the potential functions, we adopt the point of view that ψ is a perturbation on ϕ in the sense that ϕ represents an irrotational flow and ψ will only be non-zero if there is vorticity or lift. For uniform incoming flow about a symmetric thin airfoil an appropriate set of far field boundary conditions for ϕ are given by

$$\phi = x \tag{14}$$

The tangency condition is imposed at $y = 0$ by

$$\phi_y = \psi_x + (\phi_x + \psi_y)\left(\frac{dy}{dx}\right)_{airfoil} \tag{15}$$

An appropriate upstream far field boundary condition for ψ is given by

$$\psi = 0 \tag{16}$$

The ψ-function behaves much like a stream function which motivates the boundary condition on the lower boundary ($y = 0$)

$$\psi = 0 \tag{17}$$

If this were a lifting airfoil problem then ψ would be chosen as some non-zero constant on the airfoil surface in order to satisfy a Kutta condition (see Section III). A shock induced rotational flow can give rise to a velocity defect in u at the right boundary so that ψ_y must be free to vary. A suitable downstream far field boundary condition to ensure that $v = 0$ for $\phi = x$ is given by

$$\psi_x = 0 \tag{18}$$

On the top boundary we set $u = 1$, and because $\phi = x$, we impose

$$\psi_y = 0 \tag{19}$$

Note that the ψ-function is zero at the upper left corner, but can be non-zero at the upper right corner provided there is vorticity in the airfoil wake. This problem is resolved by a limiting process on the upper boundary where

$$\lim_{y \to \infty} \int_{-\infty}^{\infty} \psi_x dx = constant \tag{20}$$

and

$$\lim_{y \to \infty} \psi_x = 0 \tag{21}$$

As the upper boundary is extended farther away from the airfoil in both the x and y-directions, the ψ-function variation will be small yet permit a change from zero at the upper left boundary to the appropriate value on the upper right boundary. This process has been numerically verified by observing that the top boundary velocities approach the uniform condition.

Equation (5) is a convective entropy equation that can be marched in the x-direction. Initial entropy data behind the shock is obtained from the Euler equations. The entropy is constant on the airfoil and symmetry streamline in the wake region.

Zone Interfacing

The two different sets of governing equations are interfaced at their zone boundaries. In Fig. 3, for example, boundary values can be applied to the conservation equations (1) along the curve $efgh$ by differentiating ϕ and ψ. Along the inner boundary curve $abcd$ one can specify s, u, and v from the solution of Eq. (1) so that

$$\begin{aligned} \phi_x + \psi_y &= u_{specified} \\ \phi_y - \psi_x &= v_{specified} \end{aligned} \tag{22}$$

supply derivative boundary conditions for ϕ and ψ. By overlapping the boundaries as illustrated, information can be efficiently transferred from one domain to the other. In overlap regions both equation sets are being solved and are differentially equivalent if the shock wave is avoided.

We have used another way to interface the two zones together. From the vantage point of computational simplicity, it is desirable to interface the equations using as little special logic as possible. Except across the shock wave, the dual potential equations are everywhere equivalent to Eq. (1) for steady flow with a uniform incoming stream. Even across the shock

Eq. (9) is valid in the sense that it perserves the correct weak solution. Consequently we can solve Eq. (9) throughout the entire domain without having to define a hole with special boundary condition treatment as discussed above. This simplifies the computer code logic, especially since we avoid solving Eq. (22) on the zonal boundary. Also, by avoiding the hole, we solve the continuity prediction equation for ϕ more implicitly. That is, we avoid iteratively lagged inner boundary data which must be updated from the solution of the conservation-law form equations.

The Crocco equation which is used as a prediction equation for ψ does not admit a Rankine-Hugoniot jump solution. Nevertheless, we have also used this equation throughout the entire domain, including across the shock so as to avoid integrating Eq. (22) at the zone boundary and to improve implicitness. However, we supply the right-hand-side vorticity function of Eq. (10) in the zone *abcd* directly from the solution of Eq. (1). That is, we solve (shown in Cartesian form)

$$\psi_{xx} + \psi_{yy} = (u_y - v_x)_{specified} \qquad (23a)$$

or

$$\psi_{xx} + \psi_{yy} = (\gamma M^2)^{-1}(vs_x - us_y)_{specified} \qquad (23b)$$

where s or derivatives of u and v are specified from the conservation-law form equations. While this equation is not strictly valid across the shock, we have used it together with Eq. (10) so as to avoid coding an inner hole boundary condition (i.e., on *abcd* in Fig. 3) and to improve implicitness.

The equation for convection of entropy, Eq. (5), cannot be used across the shock. Consequently, in the entropy production zone, here bounded by the curve *abcda* of Fig. 3, we update entropy from the solution of Eq. (1) using the thermodynamic relation

$$s = (\gamma - 1)^{-1} \ell n[\gamma(\gamma - 1)M_\infty^2(e - \frac{1}{2}\rho q^2)/\rho^\gamma] \qquad (24)$$

Summarizing these concepts, we have opted for the following interface scheme. For the vorticity production zone enclosed by the curve *efghe* shown in Fig. 3, Eq. (1) is solved. Values of ρ, ρu, ρv, and e are supplied on the outer boundary of this zone from the solution of the dual potential equations. The dual potential equations, Eqs. (9) and (10), are solved over the entire flow field domain. However, within the vorticity zone circumscribed by *abcda*, Eq. (23) is used in place of Eq. (10). This equation substitution or modification is readily accomplished with simple flags and only one basic set of computer algorithms are needed to solve the dual potential equations over the entire domain. In this same zone Eq. (24) replaces Eq. (5).

III. Circulation Treatment

In a code based solely on a scalar potential it is necessary to build cuts into the field in order to treat flow with circulation. Across these cuts the scalar (i.e., velocity) potential is discontinuous and the difference equations must be specially coded to account for this jump. Because the dual potential equations allow vorticity to convect through the flow field, it is no longer necessary to build such circulation cuts into the field. This simplicity partially compensates for having to solve Eq. (10).

In order to treat lifting airfoils with the dual potential equations we adjust ψ on the airfoil so as to satisfy the Kutta condition. In essence we handle the vector potential component ψ much as if it were the stream function. When solving a flow with a stream function formulation, the value of the stream function on the body is a constant which is determined by the Kutta condition. Here we implement the Kutta condition by requiring that the magnitude of the trailing edge velocities match on the upper and lower surfaces (see Fig. 4). That is

$$q_{upper} = q_{lower} \qquad (25)$$

or

$$\left(\frac{\eta_y u - \eta_x v}{\sqrt{\eta_x^2 + \eta_y^2}}\right)_{upper} = -\left(\frac{\eta_y u - \eta_x v}{\sqrt{\eta_x^2 + \eta_y^2}}\right)_{lower} \qquad (26)$$

For isentropic flows this is equivalent to $p_{upper} = p_{lower}$. The Cartesian velocity components u and v can be eliminated in terms of ϕ and ψ as given by Eq. (12). Making use of the boundary condition that ψ is constant on the body ($\psi_\xi = 0$), Eq. (26) becomes

$$\left[\sqrt{\frac{J}{\alpha_3}}(\phi_\xi + \alpha_3\psi_\eta)\right]_{upper} = -\left[\sqrt{\frac{J}{\alpha_3}}(\phi_\xi + \alpha_3\psi_\eta)\right]_{lower} \qquad (27)$$

This relation can be differenced to provide the value of ψ on the body that satisfies the Kutta condition.

IV. Numerical Algorithms

Implicit, approximately factored (AF) finite difference schemes are used to solve the governing equations. A semi-flux split algorithm is used for Eq. (1), and alternating direction methods are used with Eqs. (11) and (12).

Semi-Flux Split Algorithm

The semi-flux split Euler equations in strong conservation-law-form, Eq. (1), are solved in delta form using the implicit approximately factored finite difference scheme

$$[I + h\nabla_\xi(\xi_x A^+) + h\delta_\eta(\eta_y B)] \times [I + h\Delta_\xi(\xi_x A^-)]\Delta\hat{Q}^n = -hR^n \qquad (28)$$

where

$$R = [\delta_\xi^b(\xi_x J^{-1}F^+) + \delta_\xi^f(\xi_x J^{-1}F^-) + \delta_\eta(\eta_y J^{-1}G)] + \epsilon J^{-1}(\nabla_\eta\Delta_\eta)^2 Q$$

$$\Delta\hat{Q} = (Q/J)^{n+1} - (Q/J)^n$$

and $F^\pm$ are flux split vectors as derived by Steger and Warming[3] while $A^\pm$ are the Jacobian matricies $\frac{\partial F^\pm}{\partial Q}$. The vector of primitive variables is

$$Q = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ e \end{pmatrix}$$

The parameter h is a constant pseudo-time step chosen to accelerate the convergence rate. All metric quantities are computed with second-order accurate difference formulas using one sided differences at the flow boundaries and central differencing interior to the flow boundaries. The difference operators in Eq. (28) are defined by

$$\nabla_\xi = ()_j - ()_{j-1}$$

$$\Delta_\xi = ()_{j+1} - ()_j$$

$$\delta_\xi^f = \frac{-3()_j + 4()_{j+1} - ()_{j+2}}{2}$$

$$\delta_\xi^b = \frac{3()_j - 4()_{j-1} + ()_{j-2}}{2}$$

$$\delta_\eta = \frac{()_{k+1} - ()_{k-1}}{2}$$

where $\Delta\xi = \Delta\eta = 1$. For simplicity, first-order accurate one sided derivative operators are used in the implicit part and second-order accurate one sided differences are used in the residual. The converged solution will be second-order accurate and unconditional linear stability is still retained. The flux splitting in the ξ-direction is inherently dissipative; however, fourth order dissipation is added in the η-direction with $\epsilon \sim O(h)$.

The first factor of Eq. (28) is solved by sweeping forward in ξ and inverting block tridiagonal systems of equations in the η-direction. The second factor forms an upper bidiagonal matrix which is solved by a simple back sweep. In supersonic regions the second factor disappears ($F^- = 0$) so the AF scheme reduces to a single direct inversion of the time-linearized equations. In our application the flow in the vorticity production zone is mostly supersonic; consequently, the AF scheme is very efficient in that zone.

Dual Potential Algorithms

The dual potential equations are weakly coupled in the sense that one can effectively solve Eq. (12) for ψ as a function of lower derivative terms that are fixed at a previous iteration level. Once ψ is predicted, Eq. (11) serves as a prediction equation for ϕ. At this point entropy can be obtained from Eq. (5) and density is updated from the Bernoulli equation.

Implicit approximately factored finite difference algorithms are used to solve the dual potential equations. We first describe the numerical algorithm for Eq. (11) which is very similar to what is used for the transonic full potential equations. Our implementation follows that of Steger and Caradonna[4]. The AF scheme is second-order accurate in subsonic flow regions and is first or second-order accurate in supersonic flow regions. Upwinding in the supersonic region is accomplished by using Hoist's upwind shifted density scheme[5].

The AF scheme for Eq. (11) is given in delta form as

$$[I - h\nabla_\eta(\rho^n n_y^2 J^{-1})_{k+1/2}\Delta_\eta] \times [I + \beta\nabla_\xi - h\nabla_\xi(\bar{\rho}^n \bar{\xi}_x^2 \bar{J}^{-1})_{j+1/2}\Delta_\xi]\Delta\phi^n = h\omega\Phi_{Rec}^n \qquad (29)$$

where

$$\Phi_{Rec} = \nabla_\xi(\bar{\rho}\bar{\xi}_x^2 \bar{J}^{-1})_{j+1/2}\Delta_\xi\phi + \nabla_\eta(\rho\eta_y^2 J^{-1})_{k+1/2}\Delta_\eta\phi + [\delta_\xi(\rho\delta_\eta\psi) - \delta_\eta(\rho\delta_\xi\psi)]$$

The j,k–indices correspond to the ξ,η–directions and are suppressed except to indicate midpoint values. The pseudo–time step h and the relaxation parameter ω are chosen so as to accelerate the iterative convergence. The ξ–difference operators are defined by

$$\delta_\xi = \frac{()_{j+1} - ()_{j-1}}{2}$$
$$\nabla_\xi = ()_j - ()_{j-1}$$
$$\Delta_\xi = ()_{j+1} - ()_j$$

where again for convenience $\Delta\xi = 1$. Similar expressions define the η–difference operators. Stability in supersonic regions is maintained by shifting the density upwind in the ξ–direction according to the formula

$$\bar{\rho}_{j+1/2} = (1-\nu_j)\left(\frac{\rho_{j+1}+\rho_j}{2}\right) + \nu_j\left[\frac{(1+\theta)\rho_j + (1-\theta)\rho_{j-1}}{2}\right]$$

with

$$\nu_j = max[0,(M_j^2-1)C]\ ,\ 1 \le C \le 2$$

At subsonic points, $\nu_j = 0$ and $\bar{\rho}_{j+1/2}$ is second–order accurate. The parameter $\theta = 2$ provides second–order accuracy at supersonic points and $\theta = 0$ gives first–order accuracy. In practice a value of $\theta = 1.8$ is used for flows with weak shocks and θ is dropped to zero for stronger shock flows. The term $\beta\nabla_\xi(\Delta\phi^n)$ provides some additional iterative damping and compensates somewhat for not properly linearizing the density term for inclusion on the left hand side of Eq. (29). The metrics are computed by the following relations in order to capture uniform flow

$$\xi_{x_j} = \frac{2}{(x_{j+1} - x_{j-1})}$$
$$\eta_{y_k} = \frac{2}{(y_{k+1} - y_{k-1})}$$
$$\eta_{y_{k+1/2}} = \frac{1}{2}(\eta_{y_{k+1}} + \eta_{y_k})$$
$$\bar{\xi}_{x_{j+1/2}} = \left(\frac{1}{x_{j+1} - x_j}\right)$$

Equation (29) is solved in the usual fashion by inverting scalar tridiagonal systems of equations in first the η and then the ξ–directions.

A second–order accurate difference for the tangency boundary condition, Eq. (15), is implemented using central differencing for x–derivatives and three point one sided differencing for the y–derivatives. A tridiagonal solution provides ϕ at the lower boundary with the ψ terms evaluated explicitly.

A similar algorithm for the vorticity equation is used to update ψ.

$$[I - h\nabla_\eta(\eta_y^2 J^{-1})_{k+1/2}\Delta_\eta] \times [I - h\nabla_\xi(\bar{\xi}_x^2 J^{-1})_{j+1/2}\Delta_\xi]\Delta\psi^n = h\omega\Psi_{Res}^n \quad (30)$$

where

$$\Psi_{Res} = \nabla_\xi(\bar{\xi}_x^2 J^{-1})_{j+1/2}\Delta_\xi\psi + \nabla_\eta(\eta_y^2 J^{-1})_{k+1/2}\Delta_\eta\psi - (\gamma M^2 J)^{-1}[v\xi_x\delta_\xi s - u\eta_y\delta_\eta s]$$

Here though the equation remains elliptic throughout. The Neumann boundary conditions Eqs. (18) and (19) provide ψ on the downstream and top boundaries using second–order accurate one sided differences.

Finally, Eq. (5) is solved for entropy. The stretched Cartesian grid used for the zonal test problem is closely aligned with the flow, so it is a simple matter to obtain an update of s by marching the equation in the ξ–direction. A finite difference scheme for marching in ξ is given by

$$\delta_\xi^b s + \left(\frac{v\eta_y}{u\xi_x}\right)\delta_\eta s = 0 \quad (31)$$

where δ_ξ^b is a three point backward difference in ξ and δ_η is a central difference in η. At each ξ–station a scalar tridiagonal inversion in η is required.

For lifting airfoils we must also satisfy the Kutta condition by differencing Eq. (27) and solving for ψ on the body. The ξ–derivatives are central differenced and the η–derivatives are forward differenced. Solving

for ψ on the body

$$\psi_{body} = (\mu^2[(a\alpha_3)_{jm,1}\psi_{jm,2} + (a\alpha_3)_{2,1}\psi_{2,2}] - (\mu-1)[(a\alpha_3)_{jm,1}\psi_{jm,3} + (a\alpha_3)_{2,1}\psi_{2,3}] + \mu\Phi)/((2\mu-1)[(a\alpha_3)_{jm,1} + (a\alpha_3)_{2,1}]) \tag{32}$$

where

$$a = \sqrt{\frac{J}{\alpha_3}}, \Phi = (a\phi_\xi)_{jm,1} + (a\phi_\xi)_{2,1}$$

$$\mu = \begin{cases} 1, & \rightarrow \text{first order} \\ 2, & \rightarrow \text{second order} \end{cases}$$

The j,k-indices correspond to the O-grid topology in Fig. 4. The Kutta condition is evaluated explicitly at the end of each iteration of Eqs. (29) and (30).

V. Results and Discussion

A nonlifting biconvex airfoil was chosen as a test problem to verify the zonal approach as an alternative to the Euler equations for solving transonic rotational flow. The subcritical flow about a NACA 0012 airfoil at angle of attack was used to demonstrate the capability of the dual potential equations to treat lift without the use of circulation cuts. All computations were performed on a CDC 7600.

The biconvex airfoil problem was computed on a stretched Cartesian grid with the small disturbance boundry condition Eq. (15) imposed at the lower boundary, y=0. Figure 5 shows the grid in the vicinity of the airfoil with the far field boundary 3 chord lengths away to either side of the airfoil and 5 chord lengths away in the y-direction. Exponential stretching was used away from the airfoil in both the x and y-directions while a spline function distributes points along the airfoil ($0 \leq x \leq 1$). This allows grid clustering at the leading edge, trailing edge and the shock.

Results for a nonlifting 10% thick biconvex airfoil at $M_\infty = .85$ are presented for the flux split, zonal and potential schemes. Figure 6 compares the flux split Cp solution with a central differenced Euler solution[6]. A monotone solution at the shock for the flux split algorithm is obtained by conservatively switching to first-order accuracy at a few points bracketing the shock. This accounts for the smearing at the foot of the shock. A comparison of the zonal Cp solution with the same Euler solution is given in Fig. 7. Except at the shock, the flux split equations are second-order accurate and require two levels of zonal boundary data in the ξ-direction. The zonal steady state solution accuracy is not sensitive to the amount of zonal overlap. Here we have used three points of overlap in both the ξ and η-directions. As shown in Fig. 7, the shock wave locations are coincident within one grid point. To ensure proper shock position it was found to be important to zero out a small value of entropy that the flux-split scheme produces before the shock; otherwise, the density predicted from the Bernoulli equation is inaccurate at the shock and the shock moves upstream two or three grid points. Figure 8 shows a comparison between the zonal and scalar potential solutions. Vorticity effects are evident by the zonal shock location upstream of the irrotational potential shock. In these calculations we have opted for first-order accuracy in supersonic regions by evaluating the shifted density with $\theta = 0$. The L_2 norm of the ϕ-residual is given in Fig. 9 for the scalar potential solution. We remark that the present computer code has not been optimized and uses constant pseudo-time steps. The convergence history for the zonal solution is given in Fig. 10. The convergence rate appears to be limited by Eq. (29). For a zonal computation in which the conservation-law zone was allocated 20% of the total number of grid points, the zonal method takes 60% less CPU time per iteration than the flux split algorithm. Moreover, the zonal algorithm only required 50% - 75% as many iterations that the flux split algorithm does for plottable accuracy.

The flow about a 12% thick biconvex airfoil at $M_\infty = .88$ generates a solution with a stronger shock. The vorticity effects are greater, and as before, the potential shock is downstream of the zonal solution, as shown in Fig. 11. The Mach contours shown in Figs. 12 and 13 correspond to the potential and zonal solutions respectively. The potential shock at the airfoil trailing edge is more oblique than the zonal shock. The velocity defect at the downstream boundary ($x = 4$) is given in Fig. 14 and demonstrates the ability of the equations to convect vorticity.

As a final example, the dual potential equations have been solved in general curvilinear coordinates ($\xi = \xi(x,y), \eta = \eta(x,y)$) for a subcritical flow over a NACA 0012 airfoil at two degrees angle of attack. A 76 x 34 O-grid was used, a portion of which is shown in the vicinity of the airfoil (Fig. 15). The far field is rectangular in shape with rounded corners. The outer boundary is 6 chord lengths from the airfoil in the x-direction and 8 chord lengths in the y-direction; nevertheless, the uniform flow boundary condition

$\psi = 0$ is imposed in the far field. The value of ψ on the body is obtained from the Kutta condition, Eq. (32), using the second-order accurate option. A Cp comparison with the potential solution TAIR[7,8] is given in Fig. 16. The solutions compare very well even though the dual potential solution is computed on a coarser grid (50% as many points). The convergence histories of the ϕ and ψ-residuals are shown in Fig. 17.

VI. CONCLUSIONS

A zonal algorithm which utilizes the conservation-law-form equations together with the dual potential equations has been developed to numerically simulate steady transonic flow of an inviscid fluid with vorticity. The dual potential equations are able to convect vorticity, and a consistant set of boundary conditions have been developed which permit vorticity at the outflow boundary. The zonal algorithm is able to capture the correct shock position. The dual potential equations require significantly less CPU time than the Euler equations since they are solved as an iteratively decoupled set of scalar equations. The test problems we have investigated showed up to 60% savings in total CPU time for a zonal solution over a semi-flux split solution of the Euler equations. Code optimization should provide additional savings.

The dual potential equations provide an alternate method of computing lifting airfoil flows. Specifying ψ on the body in order to satisfy the Kutta condition eliminates the need for circulation cuts; however, a Poisson equation must be solved. The increase in computational work is perhaps offset by coding simplicity. This will be especially true in a multi-element problem.

ACKNOWLEDGEMENTS

This work was jointly supported under Air Force Flight Dynamics Contract F33615-81-K-3020and NASA Ames Research Center Contract NCA2-OR745-205.

REFERENCES

[1]Hafez, M. and Lovell, D., "Transonic Small Disturbance Calculations Including Entropy Corrections," Numerical And Physical Aspects Of Aerodynamic Flows Conference Proceedings, Long Beach, California, January 1981.

[2]Sokhey, J. S., "Transonic Flow Around Axisymmetric Inlets Including Rotational Flow Effects," AIAA 18th Aerospace Sciences Meeting, January 1980.

[3]Steger, J. L. and Warming, R. F., "Flux Vector Splitting of the Inviscid Gasdynamic Equations with Application to Finite-Difference Methods," **Journal of Computational Physics**, Vol. 40, April 1981, pp. 263-293.

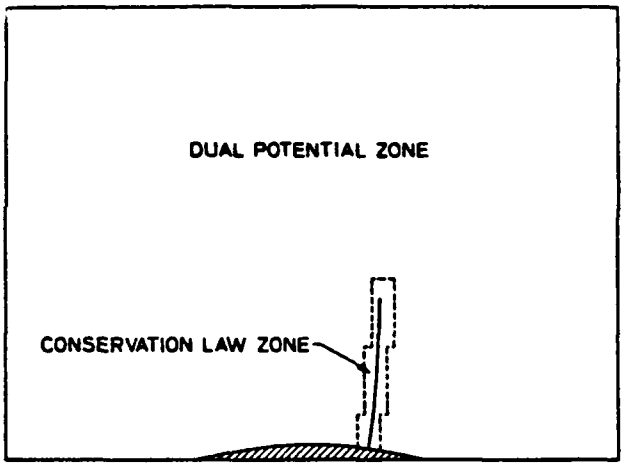
[4]Steger, J. L. and Caradonna, F. X., "A Conservative Implicit Finite Difference Algorithm for the Unsteady Transonic Full Potential Equation," AIAA 13th Fluid and Plasma Dynamics Conference, July 1980.

[5]Holst, T. L., "A Fast, Conservative Algorithm for Solving the Transonic Full-Potential Equation," AIAA Fourth Computational Fluid Dynamics Conference Proceedings, July 1979, pp. 109-121.

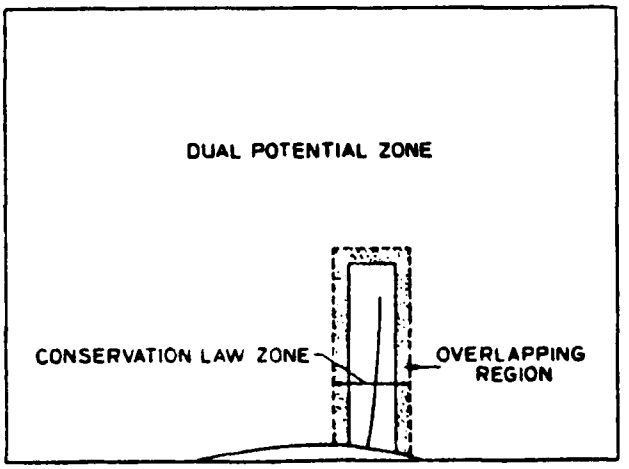
[6]Pulliam, T. H., Jespersen, D. C., and Childs, R. E., "An Enhanced Version of an Implicit Code for the Euler Equations," AIAA Paper 83-0344, AIAA 21st Aerospace Sciences Meeting, Reno, Nevada, January 1983.

[7]Holst, T. L., "An Implicit Algorithm for the Conservative, Transonic Full Potential Equation Using an Arbitrary Mesh," **AIAA Journal**, Vol. 17, October 1979, pp. 1038-1045.

[8]Dougherty, F. C., Holst, T. L., Gundy, K. L., and Thomas, S. D., "TAIR-a Transonic Airfoil Analysis Computer Code," NASA TM-81296, May 1981.



a) Small interactive zone about a shock



b) Large stationary zone about a shock

Fig. 1 Zonal partitioning.

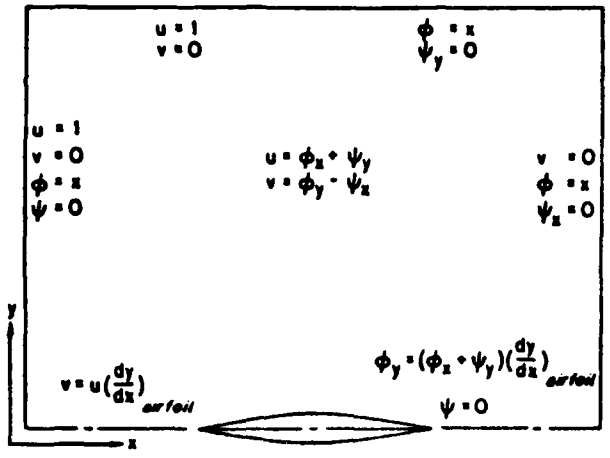


Fig. 2 Dual potential boundary conditions for a symmetric airfoil.

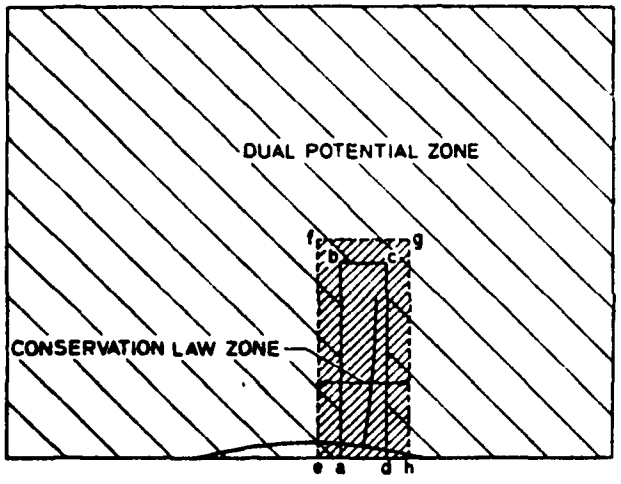


Fig. 3 Zonal boundary condition interfacing.

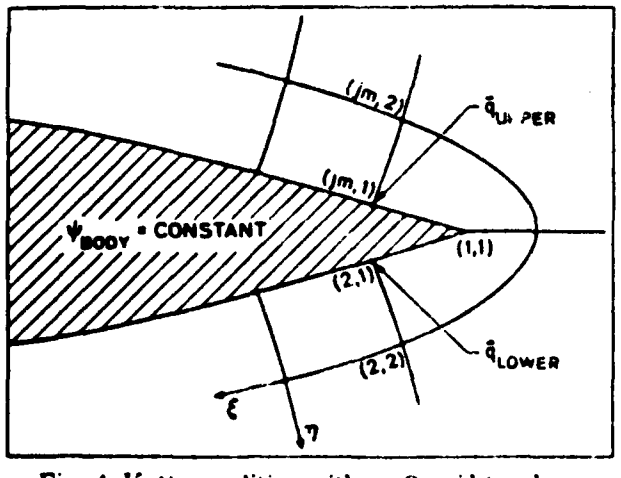


Fig. 4 Kutta condition with an O-grid topology.

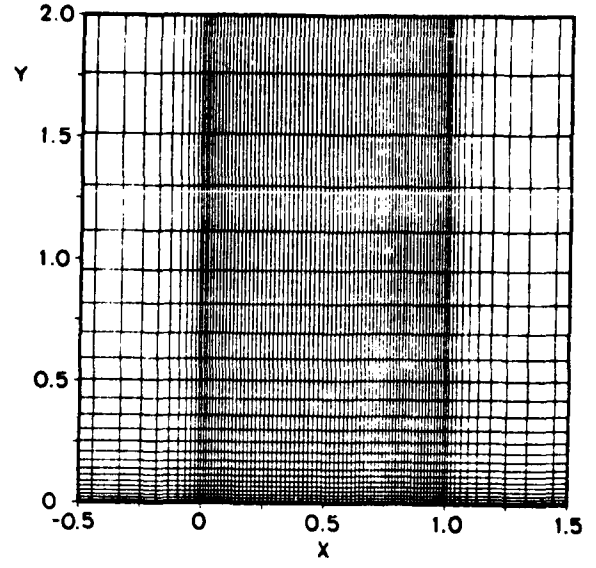


Fig. 5 Stretched Cartesian grid near airfoil.

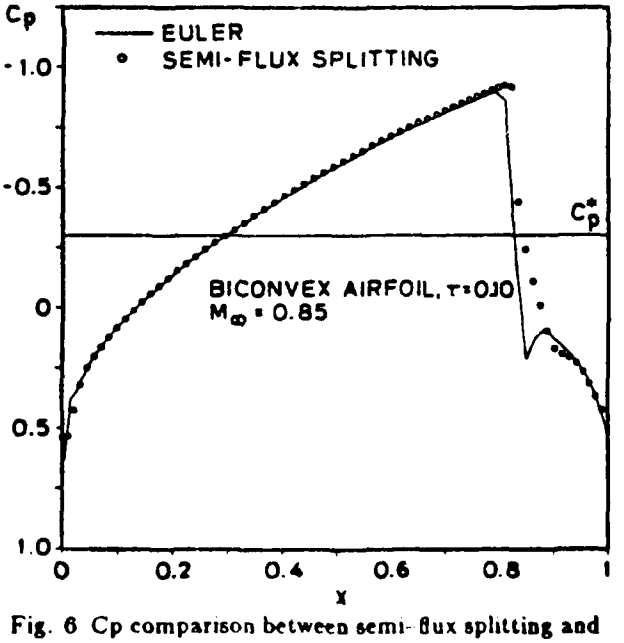


Fig. 6 Cp comparison between semi-flux splitting and centrally differenced Euler scheme.

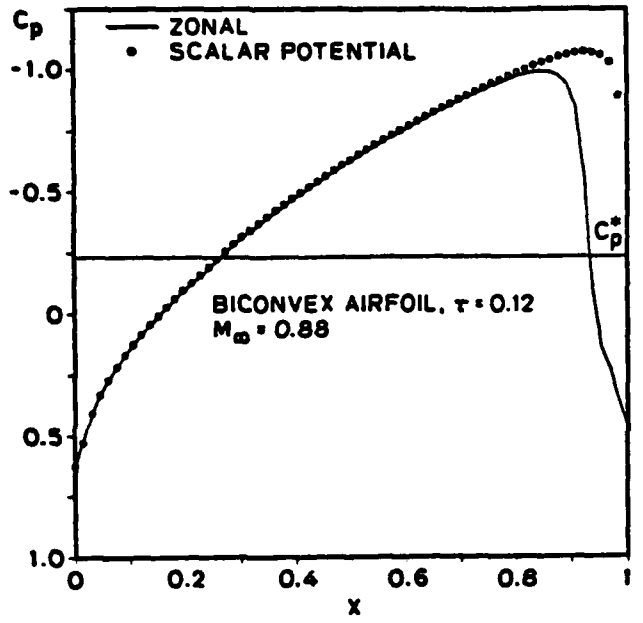


Fig. 11 Cp comparison between zonal and scalar potential scheme.

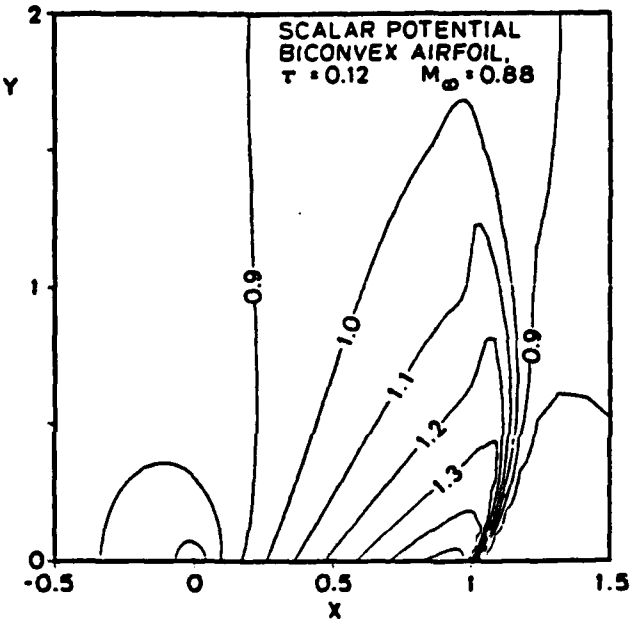


Fig. 12 Mach contours of scalar potential solution.

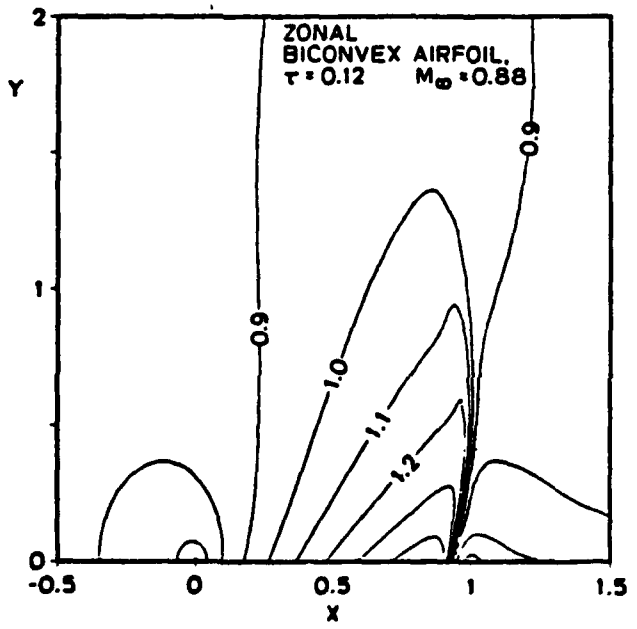


Fig. 13 Mach contours of zonal solution.

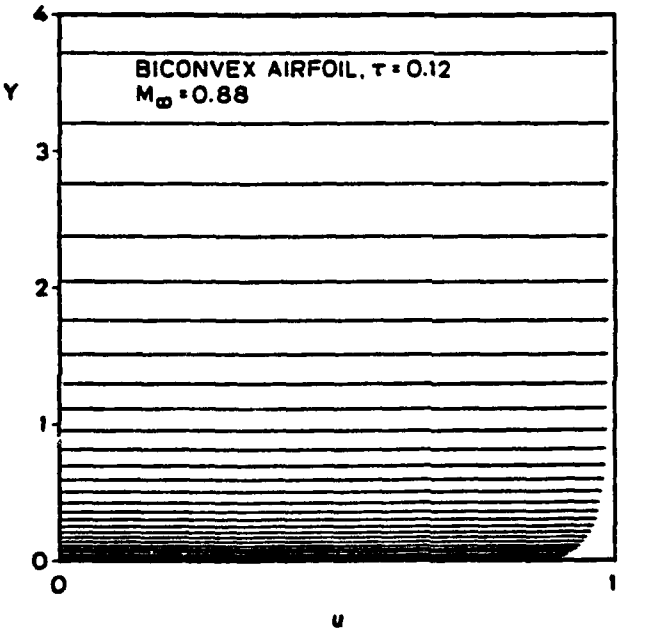


Fig. 14 Velocity vectors of zonal solution at outflow boundary (x=4).

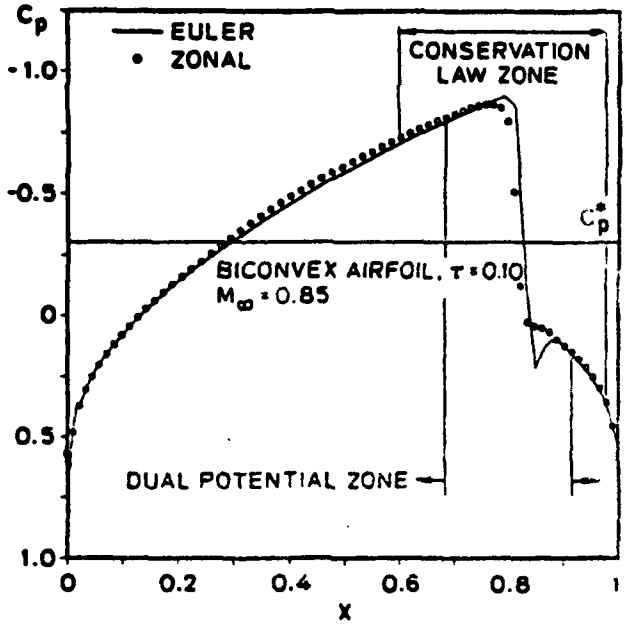


Fig. 7 Cp comparison between zonal and centrally differenced Euler scheme.

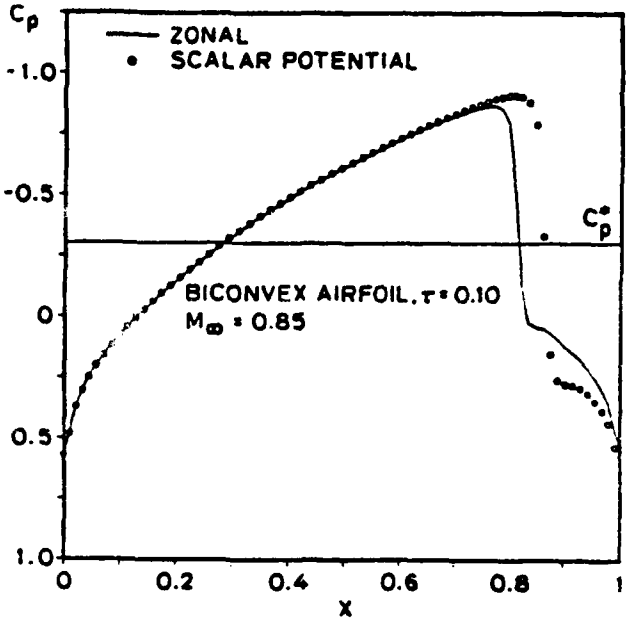


Fig. 8 Cp comparison between zonal and scalar potential scheme.

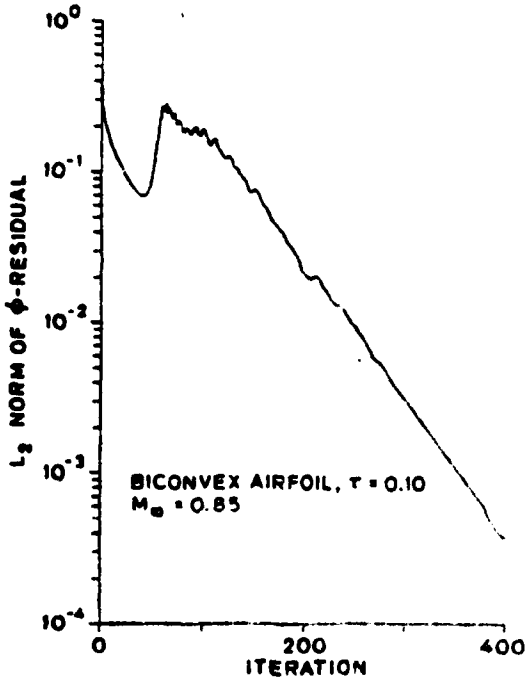


Fig. 9 Convergence history of scalar potential solution.

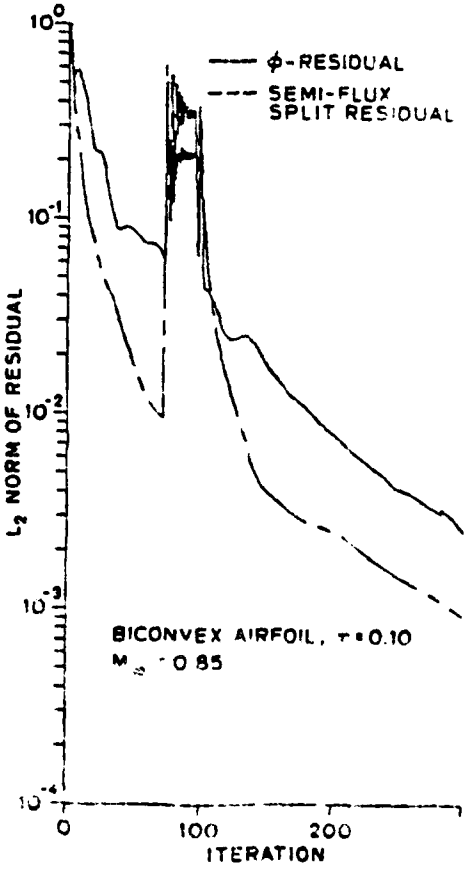
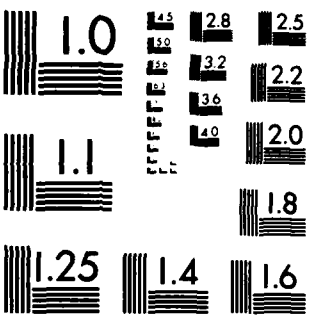


Fig. 10 Convergence histories of zonal solution.

AD-A141 692 ALGORITHMS FOR ZONAL METHODS AND DEVELOPMENT OF THREE DIMENSIONAL MESH GE..(U) STANFORD UNIV CA DEPT OF AERONAUTICS AND ASTRONAUTICS J L STEGER FEB 84

2/2

UNCLASSIFIED AFWAL-TR-83-3129 F33615-81-K-3020 F/G 12/1 NI



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

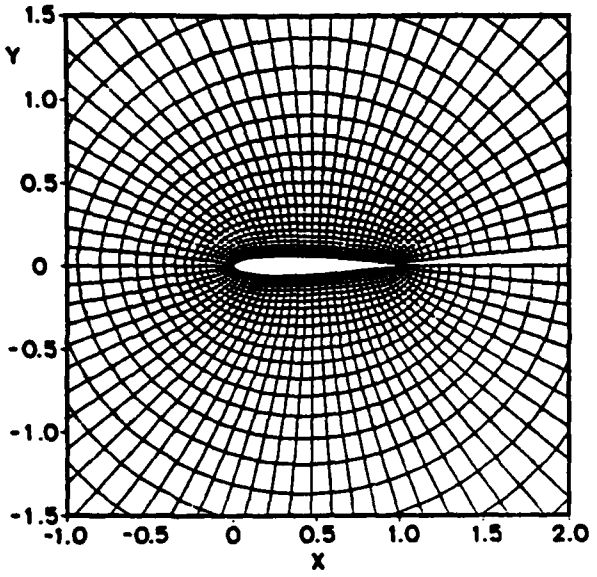


Fig. 15 O-grid topology near NACA 0012 airfoil.

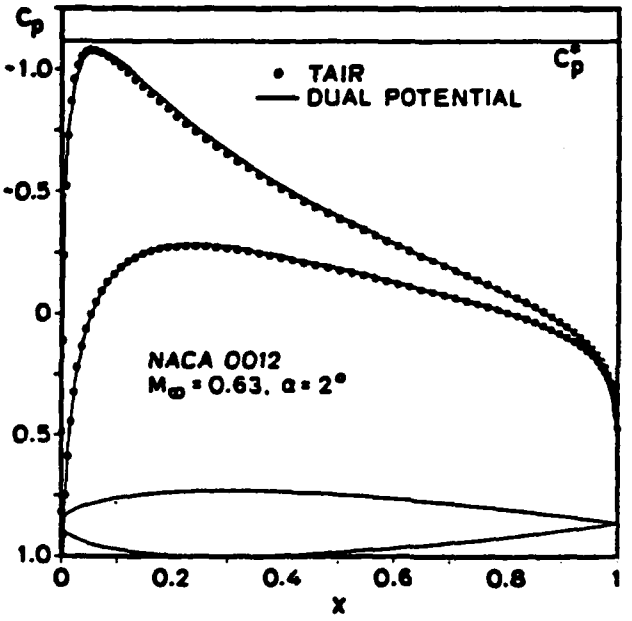


Fig. 16 Lifting subcritical Cp comparison between dual potential and scalar potential scheme.

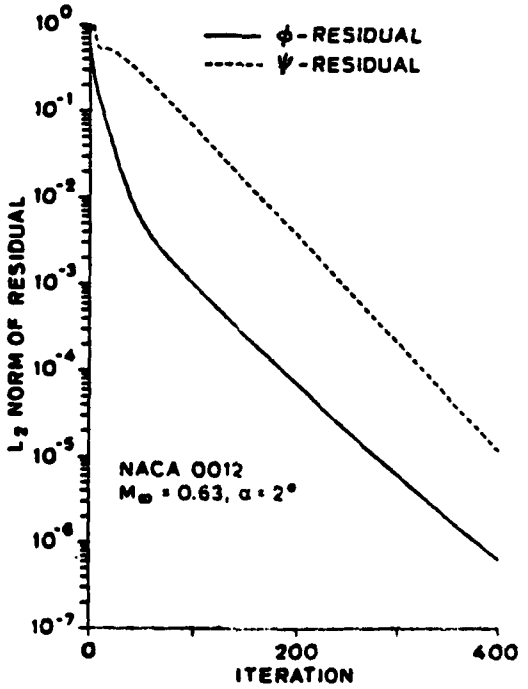


Fig. 17 Convergence histories of dual potential solution.

*USGPO: 1984-759-062-840